目录

第 1 章

信号处理基本理论

在这一章中，简要地介绍信号、系统、傅里叶分析与变换、Z 变换、滤波器结构及滤波器设计等内容。在第 3 章的相应节中，给出了利用 MATLAB 进行分析与设计的示例。

1.1 离散信号与系统

在数字信号处理(DSP)中，所有的信号都是离散(时间)信号，因此首先应解决在 MATLAB 中如何表示离散信号。

设一模拟信号经 A/D 变换后，得到序列信号

$$x(n) = \{x(n)\} = \{\cdots, x(-1), x(0), x(1), \cdots\}$$

由于 MATLAB 对下标的约定为从 1 开始递增，因此要表示 x(n)，一般应采用两个矢量，如

$$n = [-3, -2, -1, 0, 1, 2, 3, 4, 5]$$
$$y = [1, -1, 3, 2, 0, 4, 5, 2, 1]$$

这表示了一个含 9 个采样点的矢量：$y(n)=\{x(-3), x(-2), x(-1), x(0), x(2), \cdots, x(5)\}$。

通常情况下，序列值从 x(0)开始，因此一个 N 点序列 $x(n)=\{x(0), x(2), \cdots, x(N-1)$可简单地表示成

$$y = [x(0), x(1), \cdots, x(N-1)]$$

这时 y 的下标为 1～N。

1.1.1 基本信号表示

1. 单位取样序列

$$\delta(n) = \begin{cases} 1 & n = 0 \\ 0 & n \neq 0 \end{cases}$$

这一函数可利用 MATLAB 的 zeros 函数实现：

```
x=zeros(1, N);
x(1)=1;
```

还可以借助于关系操作符实现：

```
n=1:N;
x=[n==1];
```

如要产生

$$\delta(n-n_0)=\begin{cases}0 & n_1\leqslant n<n_0\\ 1 & n=n_0\\ 0 & n_0<n\leqslant n_2\end{cases}\qquad (n_1<n_2)$$

则可采用 MATLAB 实现：

```
n=n1:n2;
x=[(n-n0)==0];
```

2. *单位阶跃序列*

$$u(n)=\begin{cases}1 & n\geqslant 0\\ 0 & n<0\end{cases}$$

这一函数可利用 MATLAB 的 ones 函数实现：

```
x=ones(1, N);
```

还可借助于关系操作符“>=”来实现。如要产生在 $n_1\leqslant n_0\leqslant n_2$ 上的单位阶跃序列

$$u(n-n_0)=\begin{cases}1 & n\geqslant n_0\\ 0 & n<n_0\end{cases}$$

则可采用 MATLAB 实现：

```
n=n1:n2;
x=[(n-n0)>=0];
```

3. *实指数序列*

$$x(n)=a^n\qquad \forall\, n;\ a\in R$$

采用 MATLAB 实现：

```
n=0:N-1;
x=a.^n;
```

4. *复指数序列*

$$x(n)=e^{(\sigma+j\omega_0)n}\qquad \forall\, n$$

采用 MATLAB 实现：

```
n=0:N-1;
x=exp((du+j*w0)*n);
```

5. *正(余)弦序列*

$$x(n)=\cos(\omega_0 n+\theta)\qquad \forall\, n$$

采用 MATLAB 实现：

```
n=0:N-1;
```

```
x=cos(w0*n+Q);
```

6. 随机序列

MATLAB 中提供了两类(伪)随机信号：

rand(1, N)产生[0, 1]上均匀分布的随机矢量；

randn(1, N)产生均值为 0，方差为 1 的高斯随机序列，也就是白噪声序列。其它分布的随机数可通过上述随机数的变换而产生。

7. 周期序列

$$x(n) = x(n+N) \qquad \forall\ n$$

例如，设 x_1 表示 x 序列中一个周期的序列，要产生 4 个周期的 x 序列，用 MATLAB 实现：

```
x=[x1 x1 x1 x1];
```

1.1.2 序列操作

1. 信号加

$$x(n) = \{x_1(n) + x_2(n)\}$$

采用 MATLAB 实现：

```
x=x1+x2;
```

注意，当 x_1 和 x_2 序列的长度不等或其位置不对应时，信号相加就不是这么简单。首先应使 x_1、x_2 具有相等的长度，然后两者对齐，最后进行相加。

2. 信号乘

$$x(n) = \{x_1(n)x_2(n)\}$$

这是一种样本对样本的相乘，也即点乘运算，在 MATLAB 中可采用 .*(数组乘法)来实现，但两序列 x_1、x_2 也应经过处理。

3. 改变比例

$$y(n) = \alpha\{x(n)\} = \{\alpha x(n)\}$$

采用 MATLAB 实现：

```
y=alpha*x;
```

4. 移位

$$y(n) = \{x(n-k)\}$$

5. 折叠

$$y(n) = \{x(-n)\}$$

它将序列 x(n)在 n=0 处倒转，在 MATLAB 中可直接用 fliplr 函数实现。

6. 取样和

$$y = \sum_{n=n_1}^{n_2} x(n)$$

它不同于信号加。采用 MATLAB 实现：

```
y=sum(x(n1:n2));
```

7. 取样积

$$y = \prod_{n=n_1}^{n_2} x(n)$$

采用 MATLAB 实现：

```
y=prod(x(n1:n2));
```

8. 信号能量

$$E_x = \sum_{n=-\infty}^{\infty} x(n)x^*(n) = \sum_{n=-\infty}^{\infty} |x(n)|^2$$

“*”表示复共轭。计算一有限长序列 x 的能量 E_x，在 MATLAB 中可有两种方法实现：

```
Ex=sum(x.*conj(x));
Ex=sum(abs(x).^2);
```

9. 信号功率

$$P_x = \frac{1}{N}\sum_{n=0}^{N-1} |x(n)|^2$$

采用 MATLAB 实现：

```
Px=sum(abs(x).^2)/N;
```

1.1.3 一些重要结论

1. 单位取样综合

任何一个序列 x(n)都可由单位取样的加权和得到，即

$$x(n) = \sum_{k=-\infty}^{\infty} x(k)\delta(n-k)$$

2. 奇偶综合

任何一个序列 x(n)都可分解成偶对称部分 $x_e(n)$和奇对称部分 $x_o(n)$，即

$$x(n) = x_e(n) + x_o(n)$$

其中 $x_e(n)=\frac{1}{2}\{x(n)+x(-n)\}$

$$x_o(n)=\frac{1}{2}\{x(n)-x(-n)\}$$

这样我们可设计一函数 evenodd，完成将任一给定序列 x(n)分解成 $x_e(n)$和 $x_o(n)$，详见第 3 章。

3. 几何序列

$$x(n) = \alpha^n \qquad n \geqslant 0$$

对此有结论：

$$\sum_{n=0}^{\infty}\alpha^n = \frac{1}{1-\alpha} \qquad |\alpha| < 1$$

$$\sum_{n=0}^{N-1}\alpha^n = \frac{1-\alpha^N}{1-\alpha} \qquad \forall\ \alpha$$

4. 序列相关

两序列 x(n)和 y(n)的相似程度由相关性决定

$$r_{xy}(l) = \sum_{n=-\infty}^{\infty} x(n)y(n-l)$$

当 y(n)=x(n)时，即可求出自相关

$$r_{xx}(l) = \sum_{n=-\infty}^{\infty} x(n)x(n-l)$$

1.1.4 离散系统

最重要的是线性时不变系统。线性时不变系统的输入输出关系可通过冲激响应 h(n)表示

$$y(n) = x(n) \circledast h(n) = \sum_{k=-\infty}^{\infty} x(k)h(n-k)$$

其中⊛表示卷积运算，MATLAB 提供了求卷积函数 conv。

线性时不变系统中涉及两个重要概念：稳定性和因果性。这些内容可参看 DSP 的经典教材。

1.1.5 卷积

求卷积

$$y(n) = x(n) \circledast h(n) = \sum_{k=0}^{N} x(k)h(n-k)$$

可直接采用 MATLAB 中的函数 conv，即

```
y=conv(x, h);
```

它默认序列从 n=0 开始。但如果序列是从一负值开始，即如

$$\{x(n);\ n_{xb} \leqslant n \leqslant n_{xe}\}$$

$$\{h(n);\ n_{hb} \leqslant n \leqslant n_{he}\}$$

其中 $n_{xb}<0$ 或 $n_{hb}<0$，或两者同时为负，这样就不能直接采用 conv 函数。通过分析，其卷积结果序列为

$$\{y(n):\ n_{yb} \leqslant n \leqslant n_{ye}\}$$

且

$$n_{yb} = n_{xb} + n_{hb}$$

$$n_{ye} = n_{xe} + n_{he}$$

这样我们就可构成一新的卷积函数 con_m，它可求出带下标的序列卷积，详见第 3 章。

1.2 离散时间傅里叶分析

线性时不变系统可用单位冲激响应 h(n)表示，其输入输出之间的关系可用卷积来表示

$$y(n) = h(n) \circledast x(n)$$

1.2.1 离散时间傅里叶变换(DFT)

设序列x(n)绝对可和，即 $\sum_{n=-\infty}^{\infty}|x(n)|<\infty$，则x(n)的离散时间傅里叶变换(DFT)为

$$X(e^{j\omega}) \triangleq \mathscr{F}[x(n)] = \sum_{n=-\infty}^{\infty} x(n)e^{-j\omega n}$$

$X(e^{j\omega})$的逆离散时间傅里叶变换(IDFT)为

$$x(n) \triangleq \mathscr{F}^{-1}[X(e^{j\omega})] = \frac{1}{2\pi}\int_{-\pi}^{\pi} X(e^{j\omega})e^{j\omega n}\,d\omega$$

$X(e^{j\omega})$为一周期序列，周期为 2π，即

$$X(e^{j\omega}) = X(e^{j(\omega+2\pi)})$$

因此，在分析序列x(n)时，只需要得到一个周期的 $X(e^{j\omega})$（一般 $\omega\in[0, 2\pi]$，或 $\omega\in[-\pi, \pi]$）。对实序列x(n)，其 $X(e^{j\omega})$为共轭对称

$$X(e^{-j\omega}) = X^*(e^{j\omega})$$

或者可表示成

$$\mathrm{Re}[X(e^{-j\omega})] = \mathrm{Re}[X(e^{j\omega})]$$
$$\mathrm{Im}[X(e^{-j\omega})] = -\mathrm{Im}[X(e^{j\omega})]$$
$$|X(e^{-j\omega})| = |X(e^{j\omega})|$$
$$\angle X(e^{-j\omega}) = -\angle X(e^{j\omega})$$

这说明在绘制 $X(e^{j\omega})$曲线时，只需要绘出半个周期的曲线，一般选 $\omega\in[0, \pi]$。

1.2.2 DFT 特性

上面已提到 $X(e^{j\omega})$的两个重要特性：周期性和对称性。这里讨论 $X(e^{j\omega})$的其它特性。

1. 线性

$$\mathscr{F}[\alpha x_1(n) + \beta x_2(n)] = \alpha\mathscr{F}[x_1(n)] + \beta\mathscr{F}[x_2(n)]$$

2. 时域移位

$$\mathscr{F}[x(n-k)] = X(e^{j\omega})e^{-j\omega k}$$

3. 频域移位

$$\mathscr{F}[x(n)e^{j\omega_0 n}] = X(e^{j(\omega-\omega_0)})$$

4. 复共轭

$$\mathscr{F}[x^*(n)] = X^*(e^{-j\omega})$$

5. 折叠

$$\mathscr{F}[x(-n)] = X(e^{-j\omega})$$

6. 实序列对称性

实序列x(n)可分解成奇偶部分

$$x(n) = x_e(n) + x_o(n)$$

则有

$$\mathscr{F}[x_e(n)] = \mathrm{Re}[X(e^{j\omega})]$$

$$\mathscr{F}[x_o(n)] = j\,\mathrm{Im}[X(e^{j\omega})]$$

7. 序列卷积

$$\mathscr{F}[x_1(n)\circledast x_2(n)] = \mathscr{F}[x_1(n)]\mathscr{F}[x_2(n)] = X_1(e^{j\omega})X_2(e^{j\omega})$$

8. 序列乘积

$$\mathscr{F}[x_1(n)x_2(n)] = \mathscr{F}[x_1(n)]\circledast\mathscr{F}[x_2(n)] = \frac{1}{2\pi}\int X_1(e^{j\theta})X_2(e^{j(\omega-\theta)})\,d\theta$$

9. 能量

序列 $x(n)$的能量 E_x 可写成

$$E_x = \sum_{n=-\infty}^{\infty}|x(n)|^2 = \frac{1}{2\pi}\int_{-\pi}^{\pi}|X(e^{j\omega})|^2\,d\omega$$

$$= \int_0^{\pi}\frac{|X(e^{j\omega})|^2}{\pi}\,d\omega$$

这就是 Parseval 定理。由此可得到序列 $x(n)$的能量谱密度

$$\Phi_x(\omega) \triangleq \frac{1}{\pi}|X(e^{j\omega})|^2$$

这样，信号 $x(n)$在$[\omega_1, \omega_2]$频带内的能量为

$$\int_{\omega_1}^{\omega_2}\Phi_x(\omega)d\omega \quad 0 \leqslant \omega_1 < \omega_2 \leqslant \pi$$

1.2.3 LTI 系统频域表示

任意线性时不变系统(LTI 系统)都可由冲激响应 $h(n)$来表示，相应地在频域中可用频率响应 $H(e^{j\omega})$来表示，它是 $h(n)$的离散傅里叶变换，即

$$H(e^{j\omega}) \triangleq \sum_{n=-\infty}^{\infty}h(n)e^{-j\omega n}$$

这样对任一输入序列 $x(n)$，其输出 $y(n)$可表示为

$$y(n) = x(n)\circledast h(n)$$

在频域可表示成简单的关系

$$Y(e^{j\omega}) = X(e^{j\omega})H(e^{j\omega})$$

由差分方程表示的 LTI 系统

$$y(n) + \sum_{l=1}^{N}a_l y(n-l) = \sum_{m=0}^{M}b_m x(n-m)$$

其频域传递函数可通过输入 $x(n)=e^{j\omega n}$时，$y(n)=H(e^{j\omega})e^{j\omega n}$求出

$$H(e^{j\omega}) = \frac{\sum_{m=0}^{M}b_m e^{-j\omega m}}{1 + \sum_{l=1}^{N}a_l e^{-j\omega l}}$$

如果在$[0, \pi]$等间隔分成 $k=0, 1, \cdots, K$，则有

$$H(e^{j\omega_k})=\frac{\sum_{m=0}^{M}b_m e^{-j\omega_k m}}{1+\sum_{l=1}^{N}a_l e^{-j\omega_k l}}\qquad k=0,1,\cdots,K$$

这样，记矢量 $b=\{b_m\}$，$a=\{a_l\}(a_0=1)$，$m=\{0,1,\cdots,M\}$，$l=\{0,1,\cdots,N\}$，$\omega=\{\omega_1\ \omega_2\cdots\ \omega_K\}$，则上式可在 MATLAB 中实现：

```
H=b*exp(-j*m'*w)./(a*exp(-j*l'*w));
```

1.2.4 模拟信号的取样和重构

模拟信号经取样后可得到数字信号，数字信号经处理后要经过重构再现模拟信号。

1. **取样**

任何绝对可积信号 $x_a(t)$，其连续时间傅里叶变换为

$$X_a(j\Omega)\triangleq\int_{-\infty}^{\infty}x_a(t)e^{-j\Omega t}dt$$

其中 Ω 为模拟频率，其逆变换为

$$x_a(t)\triangleq\frac{1}{2\pi}\int_{-\infty}^{\infty}X_a(j\Omega)e^{j\Omega t}d\Omega$$

以时间间隔 T_s 对 $x_a(t)$进行取样，可得

$$x(n)=x_a(nT_s)$$

其 DFT 记为 $X(e^{j\omega})$，则有关系

$$X(e^{j\omega})=\frac{1}{T_s}\sum_{n=-\infty}^{\infty}X_a\left[j\left(\frac{\omega}{T_s}-\frac{2\pi}{T_s}l\right)\right]$$

这就是混叠方程。模拟频率 Ω 与数字频率 ω 之间的关系为

$$\omega=\Omega T_s$$

取样频率 F_s 为

$$F_s=\frac{1}{T_s}$$

对带限信号 $x(t)$，即 $X(e^{j\omega})$满足

$$X(e^{j\omega})=\frac{1}{T_s}X\left(j\frac{\omega}{T_s}\right)\qquad -\frac{\pi}{T_s}<\frac{\omega}{T_s}\leqslant\frac{\pi}{T_s}$$

时，有一个重要的取样定理。

定理 1.1(取样定理) 设 $x_a(t)$为带限信号，带宽为 F_0，则当采样频率 $F_s>2F_0$ 时，可从取样序列 $x(n)=x_a(nT_s)$中重构 $x_a(t)$，否则将导致 $x(n)$的混叠现象。带限信号的 $2F_0$ 的取样率称为 Nyquist(奈奎斯特)速率。

2. **重构**

从取样信号 $x(n)$重构原信号 $x_a(t)$是一个重要的问题。理想情况下，序列经 $F_s>F_0$(奈奎斯特速率)采样处理后，经理想的低通滤波器(截止频率为 F_s)后，可重构出 $x_a(t)$。

这时采用的内插公式为

$$x_a(t) = \sum_{n=-\infty}^{\infty} x(n)\,\mathrm{sinc}[F_s(t-nT_s)]$$

但由于它是非因果序列，所以也是不可实现的。因此采用以下几种方法：

- 零阶保持内插：使取样值在一个取样间隔内保持不变，这种滤波器的冲激响应为

$$h_0(t) = \begin{cases} 1 & 0 \leqslant t \leqslant T_s \\ 0 & \text{其它} \end{cases}$$

- 一阶保持内插：相邻取样之间连成直线，这一滤波器的冲激响应为

$$h_1(t) = \begin{cases} 1+\dfrac{t}{T} & 0 \leqslant t \leqslant T_s \\ 1-\dfrac{t}{T} & T_s \leqslant t \leqslant 2T_s \\ 0 & \text{其它} \end{cases}$$

- 三次样条内插：

$$x_a(t) = \alpha_0(n) + \alpha_1(n)(t-nT_s) + \alpha_2(n)(t-nT_s)^2 + \alpha_3(n)(t-nT_s)^3$$

$$nT_s \leqslant n < (n+1)T_s$$

其中$\{\alpha_i(n),\ 0\leqslant i\leqslant 3\}$为多项式系数，它可采用最小二乘法得到。

1.3 Z 变 换

1.3.1 双边 Z 变换

序列 x(n)的 Z 变换定义为

$$X(z) \triangleq \mathscr{Z}[x(n)] = \sum_{n=-\infty}^{\infty} x(n)z^{-n}$$

其中 z 为复变量。X(z)存在 z 的集合称为收敛域(ROC)，一般为

$$R_{x-} < |z| < R_{x-}$$

X(z)的逆 Z 变换定义为

$$x(n) \triangleq \mathscr{Z}^{-1}[X(z)] = \frac{1}{2\pi j}\oint_C X(z)z^{n-1}dz$$

其中 C 为 ROC 中包含原点的逆时针围线。

由于 ROC 是由|z|定义的，因此 ROC 一般为环形，R_{x-}可等于零，R_{x+}可为$+\infty$。当$R_{x+}<R_{x-}$时，ROC 为空，故 Z 变换不存在。

$|z|=1$(或 $z=e^{j\omega}$)称为 z 平面上的单位圆，当 ROC 包含单位圆时，则可在单位圆上计算 X(z)，实际上它就是傅里叶变换 $X(e^{j\omega})$

$$X(z)|_{z=e^{j\omega}} = X(e^{j\omega}) = \sum_{n=-\infty}^{\infty} x(n)e^{-j\omega n} = \mathscr{F}[x(n)]$$

因此，DFT 可看作 Z 变换的特殊情况。

在 Z 变换中，ROC 是它的一个重要特性，序列 Z 变换的 ROC 具有一些特点：

(1) ROC 的边界是圆；

(2) 右边序列的 ROC 为 $|z|>R_{x-}$；

(3) 左边序列的 ROC 为 $|z|<R_{x+}$；

(4) 双边带序列的 ROC 为 $R_{x-}<|z|<R_{x+}$（要是存在的话）；

(5) 序列 x(n)

$$x(n)=\begin{cases}x_n & n_1\leqslant n<n_2\\ 0 & \text{其它}\end{cases}$$

称为有限长序列，其 ROC 为整个 z 平面；

(6) ROC 中不包含极点，即以极点为界；

(7) ROC 为一连续区域。

1.3.2 Z 变换重要特性

Z 变换的特性是傅里叶变换的推广。

1. 线性

$$\mathscr{Z}[a_1x_1(n)+a_2x_2(n)]=a_1X_1(z)+a_2X_2(z)\qquad ROC:ROC_{x_1}\cap ROC_{x_2}$$

2. 取样移位

$$\mathscr{Z}[x(n-n_0)]=z^{-n_0}X(z)\qquad ROC:ROC_x$$

3. 频率移位

$$\mathscr{Z}[a^nx(n)]=X\left(\frac{z}{a}\right)\qquad ROC:ROC_x/|a|$$

4. 折叠

$$\mathscr{Z}[x(-n)]=X\left(\frac{1}{z}\right)\qquad ROC\text{：与 }ROC_x\text{ 相反，即 }ROC_x\text{ 的逆}$$

5. 复共轭

$$\mathscr{Z}[x^*(n)]=X^*(z^*)\qquad ROC:ROC_x$$

6. z 域微分

$$\mathscr{Z}[nx(n)]=-z\frac{dX(z)}{dz}\qquad ROC:ROC_x$$

7. 序列相乘

$$\mathscr{Z}[x_1(n)x_2(n)]=\frac{1}{2\pi j}\oint_C X_1(v)X_2\left(\frac{z}{v}\right)v^{-1}dv\qquad ROC:ROC_{x_1}\cap ROC_{x_2}\text{ 的逆}$$

其中 C 为公共 ROC 中包围原点的闭合围线。

8. 序列卷积

$$\mathscr{Z}[x_1(n)\circledast x_2(n)]=X_1(z)X_2(z)\qquad ROC:ROC_{x_1}\cap ROC_{x_2}$$

表 1.1 中给出一些常见序列的 Z 变换。

表 1.1 常见序列的 Z 变换

序列	Z 变换	收敛域(ROC)
$\delta(n)$	1	$0\leqslant\|z\|\leqslant\infty$
$u(n)$	$\dfrac{z}{z-1}$	$1<\|z\|\leqslant\infty$
$a^n u(n)$	$\dfrac{z}{z-a}$	$\|a\|<\|z\|\leqslant\infty$
$-a^n u(-n-1)$	$\dfrac{z}{z-a}$	$\|z\|<\|a\|$
$nu(n)$	$\dfrac{z}{(z-1)^2}$	$1<\|z\|\leqslant\infty$
$na^n u(n)$	$\dfrac{az}{(z-a)^2}$	$\|a\|<\|z\|\leqslant\infty$
$\dfrac{n(n-1)}{2!}u(n)$	$\dfrac{z}{(z-1)^3}$	$1<\|z\|\leqslant\infty$
$\dfrac{n(n-1)(n-2)}{3!}u(n)$	$\dfrac{z}{(z-1)^4}$	$1<\|z\|\leqslant\infty$
$\dfrac{(n+1)(n+2)}{2!}a^n u(n)$	$\dfrac{z^3}{(z-a)^3}$	$\|a\|<\|z\|\leqslant\infty$
$\dfrac{(n+1)(n+2)(n+3)}{3!}$	$\dfrac{z^4}{(z-a)^4}$	$\|a\|<\|z\|\leqslant\infty$
$R_N(n)$	$\dfrac{1-z^{-N}}{1-z^{-1}}$	$0<\|z\|\leqslant\infty$
$e^{j\omega_0 n}u(n)$	$\dfrac{z}{z-e^{j\omega_0}}$	$1<\|z\|\leqslant\infty$
$\sin(\omega_0 n)u(n)$	$\dfrac{z\sin\omega_0}{z^2-2\cos(\omega_0)z+1}$	$1<\|z\|\leqslant\infty$
$\cos(\omega_0 n)u(n)$	$\dfrac{z^2-\cos(\omega_0)z}{z^2-2\cos(\omega_0)z+1}$	$1<z\leqslant\infty$
$e^{-an}\sin(\omega_0 n)u(n)$	$\dfrac{e^{-a}\sin(\omega_0)z}{z^2-2e^{-a}\cos(\omega_0)z+e^{-2a}}$	$e^{-a}<\|z\|\leqslant\infty$
$e^{-an}\cos(\omega_0 n)u(n)$	$\dfrac{z^2-e^{-a}\cos(\omega_0)z}{z^2-2e^{-a}\cos(\omega_0)z+e^{-2a}}$	$e^{-a}<\|z\|\leqslant\infty$

1.3.3 逆 Z 变换

计算逆 Z 变换需计算复围线积分，这是一个复杂的过程。最切合实际的方法是部分分式展开，但要求 Z 变换必须是有理函数，这一般都是满足的。

给定

$$X(z)=\frac{b_0+b_1z^{-1}+\cdots+b_Mz^{-M}}{1+a_1z^{-1}+\cdots+a_Nz^{-N}}\qquad R_{x-}<|z|<R_{x+}$$

则求逆 Z 变换可由下列步骤完成：

(1) 重写 X(z)

$$X(z)=\frac{\tilde{b}_0+\tilde{b}_1z^{-1}+\cdots+\tilde{b}_{N-1}z^{-(N-1)}}{1+a_1z^{-1}+\cdots+a_Nz_{-N}}+\sum_{k=0}^{M-N}c_kz^{-k}$$

当 M<N 时，上式右边第二项为 0，这一项称为多项式部分，第一项称为适当有理分式部分。这种分解可采用去卷积函数(deconv)实现。

(2) 对适当有理分式部分进行部分分式展开，当 X(z)只含单重极点时有

$$X(z)=\sum_{k=1}^{N}\frac{R_k}{1-P_kz^{-1}}+\sum_{k=0}^{M-N}c_kz^{-k}$$

其中 P_k 为极点，R_k 为相应的留数。

$$R_k=\frac{\tilde{b}_0+\tilde{b}_1z^{-1}+\cdots+\tilde{b}_{N-1}z^{-(N-1)}}{1+a_1z^{-1}+\cdots+a_Nz^{-N}}(1-P_kz^{-1})\Big|_{z=P_k}$$

当 X(z)含多重极点时，设 P_k 为 r 重极点，则相应的 P_k 项变成

$$\sum_{l=1}^{r}\frac{R_{k,l}z^{-(l-1)}}{(1-P_kz^{-1})^l}=\frac{R_{k,1}}{1-P_kz^{-1}}+\frac{R_{k,2}z^{-1}}{(1-P_kz^{-1})^2}+\cdots+\frac{R_{k,l}z^{-(r-1)}}{(1-P_kz^{-1})^r}$$

(3) 写出 x(n)的形式(设只含单重极点，且 M≥N)

$$x(n)=\sum_{k=1}^{N}R_k\mathscr{Z}^{-1}\left[\frac{1}{1-P_kz^{-1}}\right]+\sum_{k=0}^{M-N}c_k\delta(n-k)$$

(4) 利用常见序列 Z 变换

$$\mathscr{Z}^{-1}\left[\frac{z}{z-P_k}\right]=\begin{cases}P_k^n u(n) & |z_k|\leqslant R_x\\ -P_k^n u(-n-1) & |z_k|\geqslant R_{x+}\end{cases}$$

代入上式，就可计算出 x(n)。

在 MATLAB 中，可直接采用 residuez 函数实现上述过程。

1.3.4 z 域系统表示

系统函数 H(z)定义为

$$H(z)=\mathscr{Z}[h(n)]=\sum_{n=-\infty}^{\infty}h(n)z^{-n}\qquad R_{h-}<|z|<R_{h+}$$

利用卷积特性，可得系统输入 X(z)与输出 Y(z)之间的关系

$$Y(z)=H(z)X(z)\quad ROC_y=ROC_h\cap ROC_x$$

以差分方程表示的系统

$$y(n)+\sum_{k=1}^{N}a_ky(n-k)=\sum_{l=0}^{M}b_lx(n-l)$$

可直接写出 H(z)

$$H(z)\triangleq\frac{Y(z)}{X(z)}=\frac{\sum_{l=0}^{M}b_lz^{-l}}{1+\sum_{k=1}^{N}a_kz^{-k}}$$

$$=\frac{b_0z^{-M}(z^M+\cdots+b_M/b_0)}{z^{-N}(z^N+\cdots+a_N)}$$

$$= b_0 z^{N-M} \frac{\prod_{l=1}^{N}(z - Z_l)}{\prod_{k=1}^{N}(z - P_k)}$$

当H(z)的ROC包含单位圆时，可计算单位圆上的H(z)，这就是$H(e^{j\omega})$，即频域传递函数

$$H(e^{j\omega}) = b_0 e^{j(N-M)\omega} \frac{\prod_{l=1}^{M}(e^{j\omega} - Z_l)}{\prod_{k=1}^{N}(e^{j\omega} - P_k)}$$

由此可给出系统的幅频和相频特性曲线。在MATLAB中可利用freqz函数直接绘出以传递函数形式表示的幅频和相频特性曲线。

稳定性是很重要的问题。对LTI系统，BIBO(有界输入有界输出)稳定性等效于$\sum_{n=-\infty}^{\infty}|h(n)| < \infty$，从DFT存在的观点出发，这种稳定性意味着$H(e^{j\omega})$存在，即单位圆$|z|=1$处在ROC中，这种结果称为z域稳定理论。

定理1.2(LTI的z域稳定)　LTI系统稳定的充要条件是H(z)的ROC包含单位圆。

定理1.3(因果LTI系统z域稳定)　因果LTI系统稳定的充要条件是H(z)的极点都在单位圆内。

1.3.5 差分方程求解

LTI系统的差分方程的解可分成通解和特解，也可分成零输入响应和零状态响应，还可看作稳态响应和暂态响应之和。为此要涉及到单边Z变换

$$\mathscr{Z}^+[x(n)] \triangleq \mathscr{Z}[x(n)u(n)] \triangleq X^+(z) = \sum_{n=0}^{\infty} x(n)z^{-n}$$

取样移位特性变成

$$\mathscr{Z}^+[x(n-k)] = x(-1)z^{1-k} + x(-2)z^{2-k} + \cdots + x(-k) = z^{-k}X^+(z)$$

将这一结果应用于差分方程

$$y(n) + \sum_{k=1}^{N} a_k y(n-k) = \sum_{m=0}^{M} b_m x(n-m) \qquad n \geqslant 0$$

设初始条件为

$$\{y(i),\ i=-1,\ \cdots,\ -N\}$$
$$\{x(i),\ i=-1,\ \cdots,\ -M\}$$

就可得到$Y^+(z)$，从而可求出y(n)。

将输出y(n)进行不同的分解，有

$$y(n) = y_g(n) + y_p(n)$$
$$y(n) = y_{tr}(n) + y_{ss}(n)$$
$$y(n) = y_{zs}(n) + y_{zi}(n)$$

其中$y_g(n)$表示通解，$y_p(n)$为特解；$y_{tr}(n)$为暂态响应，$y_{ss}(n)$为稳态响应；$y_{zs}(n)$为零状态响应，$y_{zi}(n)$为零输入响应。

为得到$y_{zs}(n)$，应将初始状态等效成输入$x_{ic}(n)$，然后利用$y_{zs}(n)=x_{ic}(n) \circledast h(n)$求解，

这可利用 MATLAB 中的 filtic 函数求出。差分方程的完全解可利用 filter 函数求得。

1.4 离散傅里叶变换

1.4.1 离散傅里叶级数

周期序列

$$\tilde{x}(n)=\tilde{x}(n+kN) \qquad \forall\, n,k$$

可表示成

$$\tilde{x}(n)=\frac{1}{N}\sum_{k=0}^{N-1}\tilde{X}(k)e^{j\frac{2\pi}{N}kn} \qquad n=0,\pm1,\cdots$$

其中 $\tilde{X}(k)$称为离散傅里叶级数的系数

$$\tilde{X}(k)=\sum_{n=0}^{N-1}\tilde{x}(n)e^{-j\frac{2\pi}{N}nk} \qquad k=0,\pm1,\cdots$$

这两个式子称为周期序列的离散傅里叶级数表示。记 $W_N\triangleq e^{-j\frac{2\pi}{N}}$，则上述两式可写成

$$\tilde{X}(k)\triangleq \mathrm{DFS}[\tilde{x}(n)]=\sum_{n=0}^{N-1}\tilde{x}(n)W_N^{nk}$$

$$\tilde{x}(n)\triangleq \mathrm{IDFS}[\tilde{X}(k)]=\frac{1}{N}\sum_{k=0}^{N-1}\tilde{X}(k)W_N^{-nk}$$

在 MATLAB 中，上述两式可利用矩阵乘法来实现，记 $\tilde{x}$ 和 $\tilde{X}$ 分别为 $\tilde{x}(n)$和 $\tilde{X}(k)$的列矢量，则有

$$\tilde{X}=W_N\tilde{x}$$

$$\tilde{x}=W_N^*\tilde{X}$$

其中

$$W_N\triangleq[W_N^{kn} \quad 0\leqslant k,n\leqslant N-1]$$

$$=\begin{bmatrix}1 & 1 & \cdots & 1\\ 1 & W_N^1 & \cdots & W_N^{(N-1)}\\ \vdots & & \vdots & \\ 1 & W_N^{(N-1)} & \cdots & W_N^{(N-1)^2}\end{bmatrix}$$

方阵 W_N 也称为 DFS 矩阵，这样可编写计算 $\tilde{x}$ 和 $\tilde{X}$ 的 MATLAB 函数，参见第 3 章的 dfs 和 idfs 函数。

DFS 与 Z 变换有直接联系，设 x(n)为有限时宽序列，即

$$x(n)=\begin{cases}\text{非零} & 0\leqslant n\leqslant N-1\\ 0 & \text{其它}\end{cases}$$

则其 Z 变换为

$$X(z)=\sum_{n=0}^{N-1}x(n)z^{-n}$$

现在我们构造一周期信号 $\tilde{x}(n)$，它是 x(n)的重复，即有

$$x(n)=\begin{cases}\tilde{x}(n) & 0\leqslant n\leqslant N-1\\ 0 & 其它\end{cases}$$

则 $\tilde{x}(n)$的 DFS 为

$$\tilde{X}(k)=\sum_{n=0}^{N-1}\tilde{x}(n)e^{-j\frac{2\pi}{N}nk}$$

比较得

$$\tilde{X}(k)=X(z)\big|_{z=e^{j\frac{2\pi}{N}k}}$$

同样 DFS 与 DFT 的关系为

$$\tilde{X}(k)=X(e^{-j\omega})\big|_{\omega=\frac{2\pi}{N}k}$$

1.4.2 z 域取样和重构

由 Z 变换和周期序列的 DFS 定义，我们可以得到任一时宽序列 $x(n)$与其周期序列 $\tilde{x}(n)$之间的关系

$$\tilde{x}(n)=\sum_{r=-\infty}^{\infty}x(n-rN)$$

这说明，当 $x(n)$为$[0,N-1]$内的时宽有限信号，则 $x(n)$可由 $\tilde{x}(n)$得以恢复

$$x(n)=\tilde{x}(n)R_N(n)$$

其中
$$R_N(n)=\begin{cases}1 & 0\leqslant n<N\\ 0 & 其它\end{cases}$$

由此可得到重要的定理。

定理 1.4(频率取样） 当 $x(n)$为$[0,N-1]$上有限时宽序列时，则 $X(z)$在单位圆上的 N 个取样可完全确定 $X(z)$。

利用这一定理，可从 $\tilde{X}(k)$的取样中恢复 $X(z)$

$$\begin{aligned}X(z)&=\mathscr{Z}[x(n)]=\mathscr{Z}[\tilde{x}(n)R_N(n)]\\&=\mathscr{Z}[\mathrm{IDFS}\{\tilde{X}(k)\}R_N(n)]\end{aligned}$$

化简后可得 z 域重构公式

$$X(z)=\frac{1-z^{-N}}{N}\sum_{k=0}^{N-1}\frac{\tilde{X}(k)}{1-W_N^{-k}z^{-1}}$$

在 z 域单位圆 $z=e^{j\omega}$上，有

$$\begin{aligned}X(e^{j\omega})&=\frac{1-e^{-j\omega N}}{N}\sum_{k=0}^{N-1}\frac{\tilde{X}(k)}{1-e^{j2\pi\frac{k}{N}}e^{-j\omega}}\\&=\sum_{k=0}^{N-1}\tilde{X}(k)\frac{1-e^{-j\omega N}}{N\{1-e^{j2\pi\frac{k}{N}}e^{-j\omega}\}}\end{aligned}$$

记内插多项式为

$$\Phi(\omega)\triangleq\frac{\sin\left(\frac{\omega N}{2}\right)}{N\sin\left(\frac{\omega}{2}\right)}e^{-j\omega\left(\frac{N-1}{2}\right)}$$

则有

$$X(e^{j\omega})=\sum_{k=0}^{N-1}\tilde{X}(k)\Phi\left(\omega-\frac{2\pi k}{N}\right)$$

这就是从 $\tilde{X}(k)$中重构 $X(e^{j\omega})$的 DFT 内插公式。

1.4.3 离散傅里叶变换

$x(n)$的 DFT 定义为

$$X(k) \triangleq \text{DFT}[x(n)] = \begin{cases} \tilde{X}(k) & 0 \leqslant k \leqslant N-1 \\ 0 & \text{其它} \end{cases}$$

或者

$$X(k) = \sum_{n=0}^{N-1} x(n) W_N^{nk} \qquad 0 \leqslant k \leqslant N-1$$

这说明 $X(k)$是 $\tilde{X}(k)$的主部，即 $X(k)=\tilde{X}(k)R_N(k)$。

IDFT 定义为

$$x(n) \triangleq \text{IDFT}[X(k)] = \tilde{x}(n)R_N(n)$$

或者

$$X(k) = \sum_{n=0}^{N-1} x(n) W_N^{nk} \qquad 0 \leqslant k \leqslant N-1$$

显然在 MATLAB 中，可很容易实现 DFT 和 IDFT：

$$X = W_N x$$

$$x = \frac{1}{N} W_N^* X$$

这与 DFS 非常类似。由此可得扩展函数 dft 和 idft，详见 3.4。

1.4.4 DFT 特性

1. 线性

$$\text{DFT}[ax_1(n) + bx_2(n)] = a\text{DFT}[x_1(n)] + b\text{DFT}[x_2(n)]$$

2. 圆形折叠

由于 N 点序列折叠后不再是 N 点序列，致使 DFT 不存在，为此在变量$-n$上采用模 N 运算，即定义

$$x((-n))_N = \begin{cases} x(0) & n = 0 \\ x(N-n) & 1 \leqslant n \leqslant N-1 \end{cases}$$

为圆形折叠。它的 DFT 为

$$\text{DFT}[x((-n))_N] = X((-k))_N = \begin{cases} X(0) & k = 0 \\ X(N-k) & 1 \leqslant k \leqslant N-1 \end{cases}$$

3. 共轭

$$\text{DFT}[x^*(n)] = X^*((-k))_N$$

4. 实序列对称性

设 $x(n)$为 N 点实序列，$x(n)=x^*(n)$，因此由特性 3 可得

$$X(k) = X^*((-k))_N$$

也即

$$\mathrm{Re}[X(k)] = \mathrm{Re}[X((-k))_N]$$
$$\mathrm{Im}[X(k)] = -\mathrm{Im}[X((N-k))_N]$$
$$|X(k)| = |X((-k))_N|$$
$$\angle X(k) = -\angle X((-k))_N$$

实序列可分解为奇偶分量

$$X_{ec} \triangleq \frac{1}{2}[x(n) + x((-n))_N] = \begin{cases} x(0) & n = 0 \\ \frac{1}{2}[x(n) + x(N-n)] & 1 \leqslant n \leqslant N+1 \end{cases}$$

$$X_{oc} \triangleq \frac{1}{2}[x(n) + x((-n))_N] = \begin{cases} 0 & n = 0 \\ \frac{1}{2}[x(n) - x(N-n)] & 1 \leqslant n \leqslant N+1 \end{cases}$$

其相应的 DFT 为

$$\mathrm{DFT}[x_{ec}(n)] = \mathrm{Re}[X(k)] = \mathrm{Re}[X((-k))_N]$$
$$\mathrm{DFT}[x_{oc}(n)] = \mathrm{Im}[X(k)] = \mathrm{Im}[X((-k))_N]$$

在 MATLAB 中，可利用 mod 函数实现实序列分解成奇偶分量。详见扩展函数 circevod（3.4 节）。

5. 序列的循环移位

x(n)的周期移位定义为

$$\tilde{x}(n-m) = x((n-m))_N$$

x(n)的循环移位定义为

$$\tilde{x}(n-m)R_N(n) = x((n-m))_N R_N(n)$$

其 DFT 为

$$\mathrm{DFT}[x((n-m))_N R_N(n)] = W_N^{km} X(k)$$

为在 MATLAB 中实现序列的循环移位，可建立扩展函数 cirshftt，详见 3.4。

6. 频域循环移位

$$\mathrm{DFT}[W_N^{-ln} x(n)] = X((k-l))_N R_N(k)$$

7. 循环卷积

循环卷积中对序列采用循环移位，其定义为

$$x_1(n) \textcircled{N} x_2(n) = \sum_{m=0}^{N-1} x_1(n) x_2((n-m))_N \quad 0 \leqslant n \leqslant N-1$$

其 DFT 为

$$\mathrm{DFT}[x_1(n) \textcircled{N} x_2(n)] = X_1(k) X_2(k)$$

在 MATLAB 中，可构造完成循环卷积的函数 circonvt，详见 3.4。

8. 序列相乘

$$\mathrm{DFT}[x_1(n) x_2(n)] = \frac{1}{N} X_1(k) \textcircled{N} X_2(k)$$

9. Parseval(帕斯维尔)关系式

$$E_x = \sum_{n=0}^{N-1} |x(n)|^2 = \frac{1}{N}\sum_{k=0}^{N-1} |X(k)|^2$$

将$|X(k)|^2/N$称为有限时宽序列x(n)的能量谱，类似地，可将$|\tilde{X}(k)|^2/N$称为周期序列的功率谱。

1.4.5 利用DFT计算线性卷积

线性系统的最重要操作之一为线性卷积，而DFT是实现线性卷积的重要方法，然而，DFT得到的是循环卷积，与线性卷积不同。

线性卷积

$$\begin{aligned} x_3(n) &= x_1(n) \circledast x_2(n) \\ &= \sum_{k=-\infty}^{\infty} x_1(k)x_2(n-k) = \sum_{k=0}^{N_1-1} x_1(k)x_2(n-k) \end{aligned}$$

设$x_1(n)$为N_1点，$x_2(n)$为N_2点，则$x_3(n)$为N_1+N_2-1点序列。

如果取$N=N_1+N_2-1$，则

$$\begin{aligned} x_4(n) &= x_1(n) \text{Ⓝ} x_2(n) \\ &= \Big[\sum_{m=0}^{N-1} x_1(k)x_2((n-m))_N\Big]R_N(n) \\ &= \Big[\sum_{r=-\infty}^{\infty} x_3(n-rN)\Big]R_N(n) \end{aligned}$$

这说明

$$x_4(n) = x_3(n) \qquad 0 \leqslant n \leqslant N-1$$

即当取$N=N_1+N_2-1$时，可利用DFT来计算线性卷积。

当$N<N_1+N_2-1$时，则采用DFT计算线性卷积时存在误差，在实际中计算这种误差是很有必要的。当采用N进行计算

$$x_4(n) = \Big[\sum_{r=-\infty}^{\infty} x_3(n-rN)\Big]R_N(n)$$

时，则有

$$e(n) \triangleq x_4(n) - x_3(n) = \Big[\sum_{r\neq 0} x_3(n-rN)\Big]R_N(n)$$

在实际操作中，需要对数据流进行分块处理，这时直接采用DFT计算线性卷积会产生一些问题，而应该将x(n)通过重复前M−1个取样来进行分块，这样可得到正确结果。

这种思路可构成通用的M函数，详见3.4的扩展函数ovrlpsav。

1.4.6 快速傅里叶变换(FFT)

N点序列x(n)的N点DFT可表示成

$$X(k) = \sum_{n=0}^{N-1} x(n)W_N^{nk} \qquad 0 \leqslant k \leqslant N-1$$

其中$W_N=e^{-j2\pi/N}$。利用系数$\{W_N^{nk}\}$的周期性

$$W_N^{kn} = W_N^{k(n+N)} = W_N^{(k+N)n}$$

和对称性

$$W_N^{kn+\frac{N}{2}} = -W_N^{kn}$$

可采用 FFT 算法来计算 x(n)的 DFT。

FFT 算法基本上分两类：时域抽取 FFT(DIT-FFT)和频域抽取 FFT(DIF-FFT)。

当 $N=2^v$ 时称为基 2 的 FFT 算法。下面先说明基 2 的 DIT-FFT 算法。

将 x(n)划分成 N/2 点序列：

$$g_1(n) = x(2n)$$

$$g_2(n) = x(2n+1) \qquad 0 \leqslant n \leqslant \frac{N}{2} - 1$$

对 $g_1(n)$和 $g_2(n)$分别经 N/2 点 DFT 得 $G_1(k)$和 $G_2(k)$，这时

$$X(k) = G_1(k) + W_N^k G_2(k) \qquad 0 \leqslant k \leqslant N-1$$

重复这一过程，可得到 x(n)的 FFT。

例如当 N=8 时，其信号流图如图 1.1 所示。

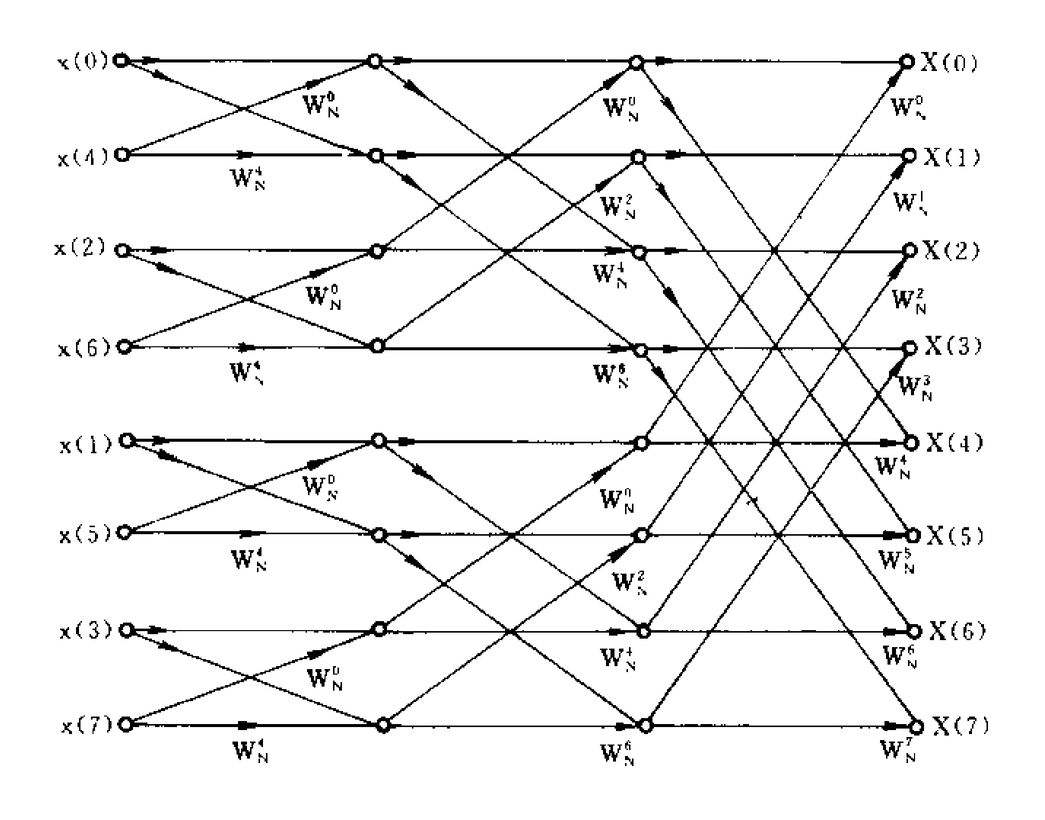


图 1.1 N=8 时 DIT-FFT 结构

基 2 的 DIF-FFT 算法只是划分方式略有差别，先将 x(n)按前后分成两个 N/2 点序列，然后继续这一过程，其信号流结构如图 1.2 所示。

在 MATLAB 中，可直接利用 fft 函数进行计算，它是由 MATLAB 系统本身提供的，因此速度快。详见第 2 章有关 fft 的函数说明。

IFFT(逆 FFT)同样可由 ifft 函数直接计算。

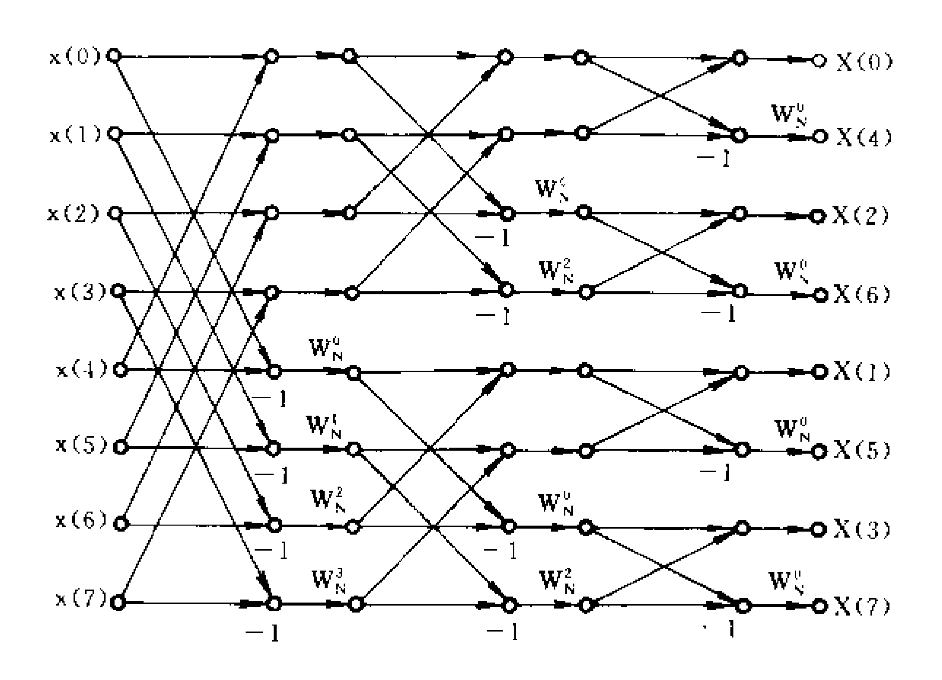


图 1.2　N=8 时 DIF-FFT 结构

1.5　数字滤波器结构

1.5.1　IIR 滤波器结构

IIR 滤波器可写成

$$H(z)=\frac{B(z)}{A(z)}=\frac{b_0+b_1z^{-1}+\cdots+b_Mz^{-M}}{1+a_1z^{-1}+\cdots+a_Nz^{-N}}$$

即可用滤波器系数 b_n 和 a_n 表示。当 $a_N\neq0$ 时，滤波器的阶为 N。IIR 滤波器的差分方程表示为

$$y(n)=\sum_{m=0}^{M}b_mx(n-m)-\sum_{m=1}^{N}a_my(n-m)$$

实现 IIR 滤波器可采用三种不同的结构：直接形式、级联形式和并联形式。

1. 直接形式

滤波器差分方程直接用延迟线、乘法器和加法器实现，例如 M=4，N=4 时，得到如图 1.3 所示的结构，其差分方程为

$$y(n)=b_0x(n)+\cdots+b_4x(n-4)-a_1y(n-1)-\cdots-a_4y(n-4)$$

这种结构称为直接形式 1。

将这种结构进行简化，可得到直接形式 2，如图 1.4 所示。

在 MATLAB 中，可利用 DSP 工具箱函数 filter 实现 IIR 的直接形式。

2. 级联形式

下面先假设 N 为偶数，则 IIR 可简化

$$H(z)=\frac{b_0+b_1z^{-1}+\cdots+b_Nz^{-N}}{1+a_1z^{-1}+\cdots+a_Nz^{-N}}$$
$$=b_0\prod_{k=1}^{K}\frac{1+B_{k,1}z^{-1}+B_{k,2}z^{-2}}{1+A_{k,1}z^{-1}+A_{k,2}z^{-2}}$$

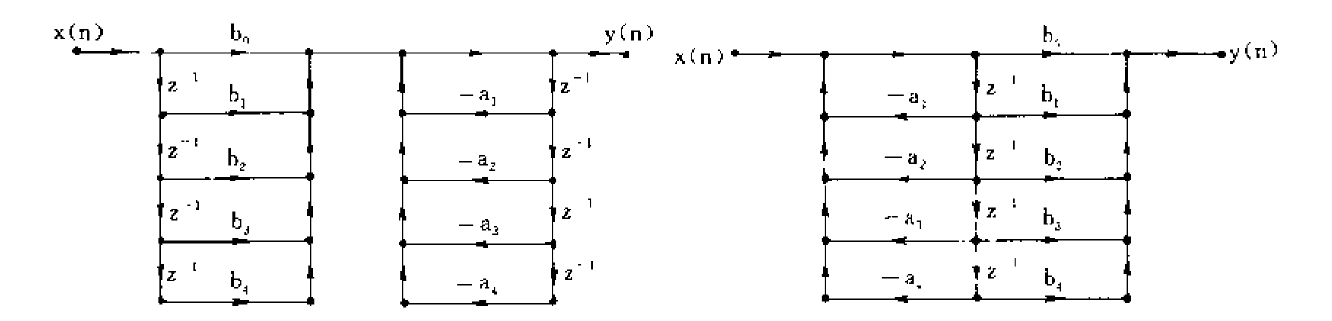


图 1.3　IIR 滤波器直接形式 1 结构(N=4)　　　图 1.4　IIR 滤波器直接形式 2 结构(N=4)

其中 K=N/2，每个二阶形式都可用直接形式 2 得以实现，例如当 N=4 时，其级联结构如图 1.5 所示。

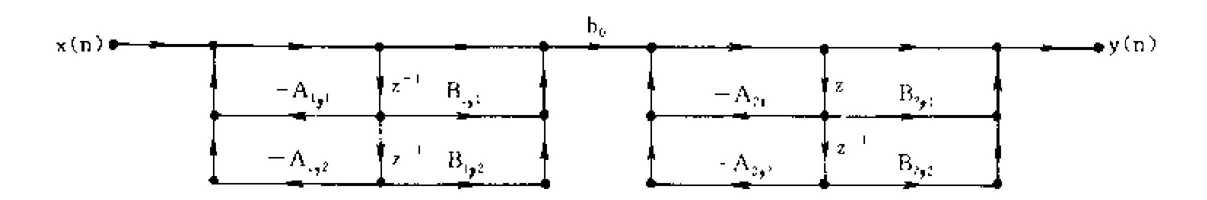


图 1.5　IIR 滤波器的级联结构(N=4)

在 MATLAB 中，给定直接形式滤波器的系数 $\{b_n\}$ 和 $\{a_n\}$，可计算出 b_0，$\{B_{k,i}\}$ 和 $\{A_{k,i}\}$，这可由扩展函数 dir2cas 实现。然后利用扩展函数 casfiltr 实现滤波器的级联形式，详见 3.5。

当然，在给定级联形式时，也可得到滤波器的直接形式，这由扩展函数 cas2dir 实现。

3. *并联形式*

在级联形式的基础上，利用部分分式展开，可进一步得到(设 M≥N)

$$H(z)=\frac{b_0+b_1z^{-1}+\cdots+b_Mz^{-M}}{1+a_1z^{-1}+\cdots+a_Nz^{-N}}$$
$$=\frac{\hat{b}_0+\hat{b}_1z^{-1}+\cdots+\hat{b}_{N-1}z^{-(N-1)}}{1+a_1z^{-1}+\cdots+a_Nz^{-N}}+\sum_{k=0}^{M-N}C_kz^{-k}$$
$$=\sum_{k=1}^{K}\frac{B_{k,0}+B_{k,1}z^{-1}}{1+A_{k,1}z^{-1}+A_{k,2}z^{-2}}+\sum_{k=0}^{M-N}C_kz^{-k}$$

其中 K=N/2，每一部分均为二阶，它们之间是并联关系。同样以 M=4，N=4 为例，画出其结构如图 1.6 所示。

在 MATLAB 下，我们设计了将滤波器直接形式转化为并联形式的扩展函数 dir2par，其中用到了复共轭对比较函数 cplxcomp。这样我们就可利用 parfiltr 函数实现滤波器的并联形式。最后还设计出了将并联形式转换成直接形式的扩展函数 par2dir。这 4 个函数可参见 3.5。

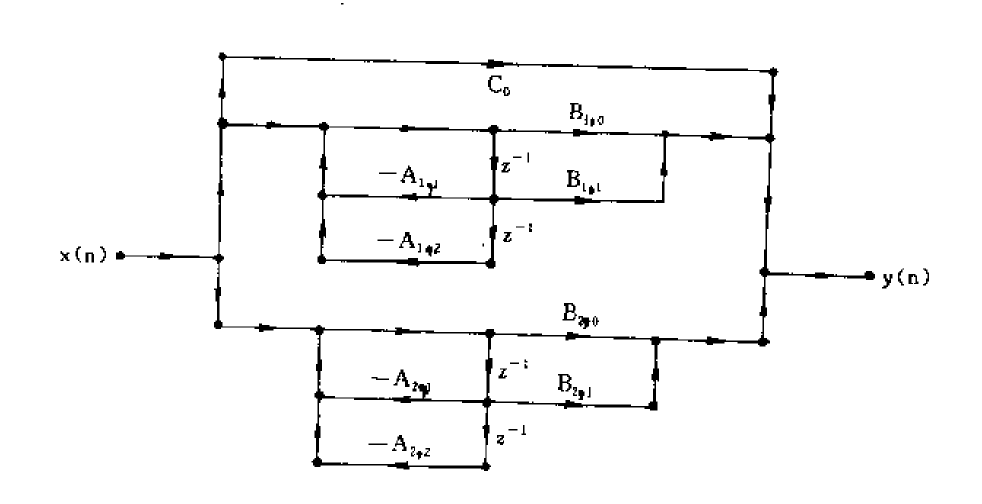


图 1.6　IIR 滤波器的并联结构(N=4)

1.5.2　FIR 滤波器结构

有限脉冲响应(FIR)滤波器可表示为

$$H(z) = b_0 + b_1 z^{-1} + \cdots + b_{M-1} z^{-(M-1)}$$

即

$$h(n) = \begin{cases} b_n & 0 \leqslant n \leqslant M-1 \\ 0 & \text{其它} \end{cases}$$

差分方程表示

$$y(n) = b_0 x(n) + b_1 x(n-1) + \cdots + b_{M-1} x(n-M+1)$$

与 IIR 滤波器相比，FIR 滤波器结构相对简单，而且可设计成线性相位。实现 FIR 滤波器可采用以下 4 种结构：直接形式、级联形式、线性相位形式和频率取样形式。

1. **直接形式**

与 IIR 类似，这是最直接的实现形式，以 M=5 为例，其结构如图 1.7 所示。在 MATLAB 中可通过 filter 函数实现。

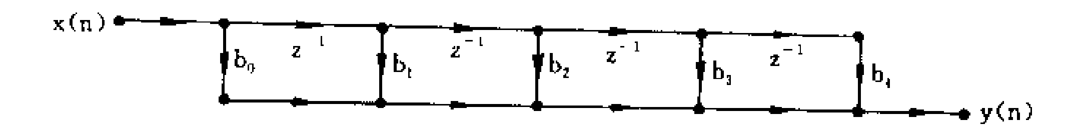


图 1.7　FIR 滤波器的直接形式(M=5)

2. **级联形式**

FIR 的级联形式类似于 IIR。

$$H(z) = b_1 + b_1 z^{-1} + \cdots + b_{M-1} z^{-(M-1)}$$
$$= b_0 \prod_{k=1}^{K} (1 + B_{k,1} z^{-1} + B_{k,2} z^{-2})$$

其中 K=M/2。以 M=7 为例，其结构如图 1.8 所示。

在 MATLAB 中，可采用 dir2cas 和 cas2dir 函数，在直接形式和级联形式之间转换，滤波器实现可通过 casfiltr 函数实现。

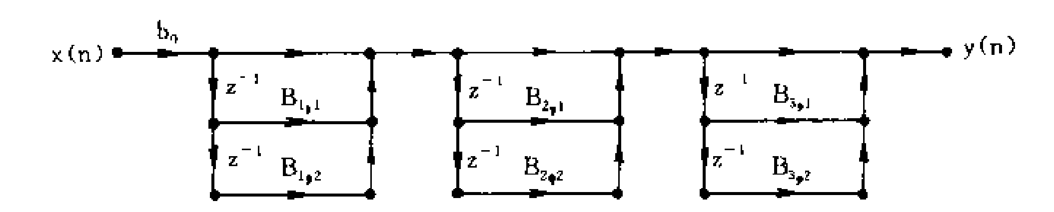


图 1.8 FIR 滤波器级联形式(M=7)

3. 线性相位形式

线性相位条件

$$\angle H(e^{j\omega}) = \beta - \alpha\omega \qquad -\pi < \omega \leqslant \pi$$

其中 $\beta=0$ 或 $\beta=\pm\pi/2$，α 为常数。对因果 FIR 滤波器，设冲激响应处于[0, M−1]区间上，则上述条件就表示下列对称性

$$h(n) = h(M-1-n) \qquad \beta = 0,\ 0 \leqslant n \leqslant M-1 \qquad (\text{对称冲激响应})$$

$$h(n) = -h(M-1-n) \quad \beta = \pm\frac{\pi}{2},\ 0 \leqslant n \leqslant M-1 \quad (\text{反对称冲激响应})$$

以 M=7 或 M=6 为例，对称型的结构如图 1.9 所示。

在 MATLAB 中，实现 FIR 线性相位形式的函数等同于直接形式，即采用 filter 函数。

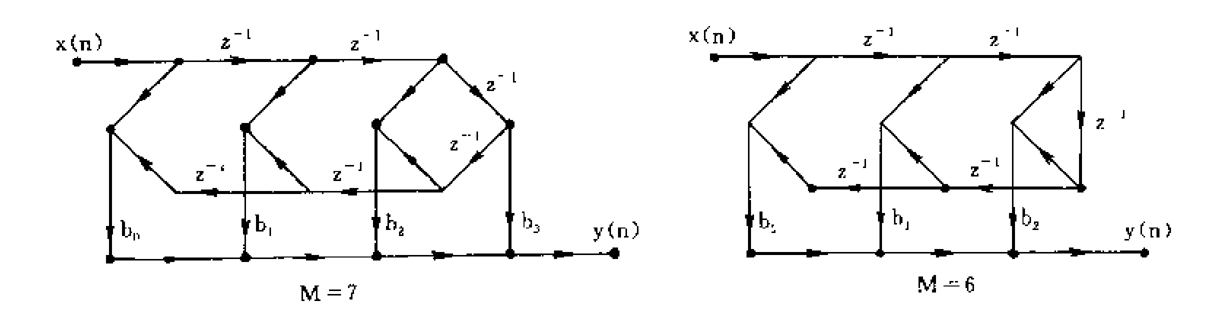


图 1.9 FIR 滤波器线性相位形式(M=7 或 M=6)

4. 频率取样形式

冲激响应 h(n)的 M 点 DFT 为 H(k)($0\leqslant k\leqslant M-1$)，则有

$$H(z) = \mathscr{Z}[h(n)] = \mathscr{Z}[\mathrm{IDFT}(H(k))]$$

利用内插公式可得

$$H(z) = \left(\frac{1-z^{-M}}{M}\right)\sum_{k=0}^{M-1}\frac{H(k)}{1-W_M^{-k}z^{-1}}$$

这表明在这种结构中采用了离散傅里叶变换 H(k)，而不是冲激响应 h(n)。这种结构分成两部分：第一部分由$(1-z^{-M})/M$ 构成，第二部分由 M 个 $H(k)/(1-W_M^{-k}z^{-1})$并联而成。以 M=4 为例，频率取样形式如图 1.10 所示。

在 MATLAB 中，给定 h(n)或 H(k)，则可借助于扩展函数 dir2fs 得到频率取样形式，它将 h(n)值转换成频率取样形式，详见 3.5。

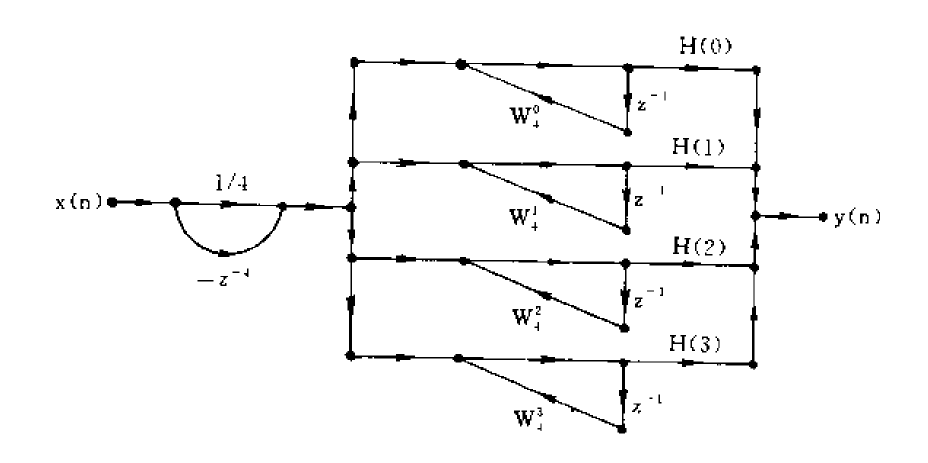


图 1.10　FIR 滤波器频率取样形式(M=4)

1.5.3　格形滤波器结构

格形滤波器在数字语言处理及自适应滤波中有着广泛的应用。它主要可分为两种：全零点格形滤波器和全极点格形滤波器。

1. 全零点格形滤波器

M−1 阶(长度为 M)FIR 滤波器具有 M−1 阶格形结构，如图 1.11 所示。

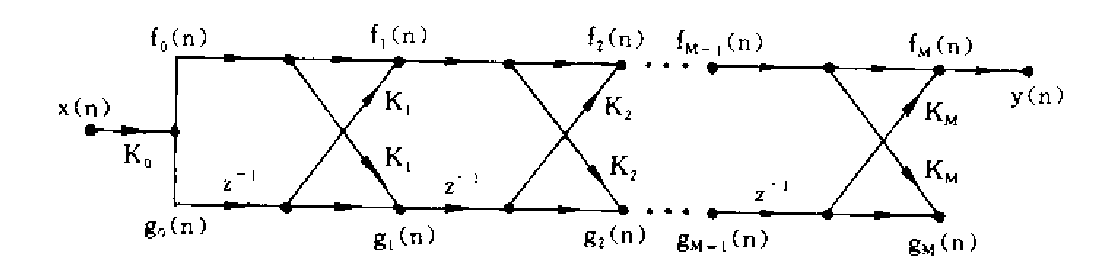


图 1.11　全零点格形滤波器

滤波器的每一阶具有简单的递推关系

$$f_m(n) = f_{m-1}(n) + K_m g_{m-1}(n-1) \quad m = 1, 2, \cdots, M-1$$

$$g_m(n) = K_m f_{m-1}(n) + g_{m-1}(n-1) \quad m = 1, 2, \cdots, M-1$$

其中 K_m 称为反射系数，初值

$$f_0(n) = g_0(n) = K_0 x(n)$$

$$y(n) = f_{M-1}(n)$$

当 FIR 以直接形式表示时

$$H(z) = \sum_{m=0}^{M-1} b_m z^{-m} = b_0\left(1 + \sum_{m=1}^{M-1} \frac{b_m}{b_0} z^{-m}\right)$$

若记多项式

$$A_{M-1}(z) = \left(1 + \sum_{m=1}^{M-1} \alpha_{M-1}(m) z^{-m}\right)$$

$$\alpha_{M-1}(m) = \frac{b_m}{b_0}$$

则格形滤波器系数$\{K_m\}$可由下列递归算法得到

$$K_0 = b_0$$
$$K_{M-1} = \alpha_{M-1}(M-1)$$
$$J_m(z) = z^{-m}A_m(z^{-1}) \qquad m = M-1, \cdots, 2, 1$$
$$A_{m-1}(z) = \frac{A_m(z) - K_m J_m(z)}{1 - K_m^2} \qquad m = M-1, \cdots, 2, 1$$
$$K_m = \alpha_m(m) \qquad m = M-2, \cdots, 2, 1$$

但应注意，一旦$|K_m|=1$，则上述算法将失效，这时线性相位 FIR 滤波器不能用格形结构实现。

在 MATLAB 中，设计出了几个有关的扩展函数，dir2latc 可将直接形式系数$\{b_n\}$转换成格形滤波器系数$\{K_m\}$，latcfilt 函数用于实现 FIR 格形滤波器，latc2dir 函数可将$\{K_m\}$转换成$\{b_n\}$，详见 3.5。

2. 全极点格形滤波器

IIR 滤波器的格形结构可实现全极点函数

$$H(z) = \frac{1}{1 + \sum_{m=1}^{N} \alpha_N(m) z^{-m}}$$

其格形结构如图 1.12 所示。

在 MATLAB 中，可利用 dir2latc、latc2dir 函数在直接形式和格形形式之间的转换，利用 latcfilt 函数实现滤波。

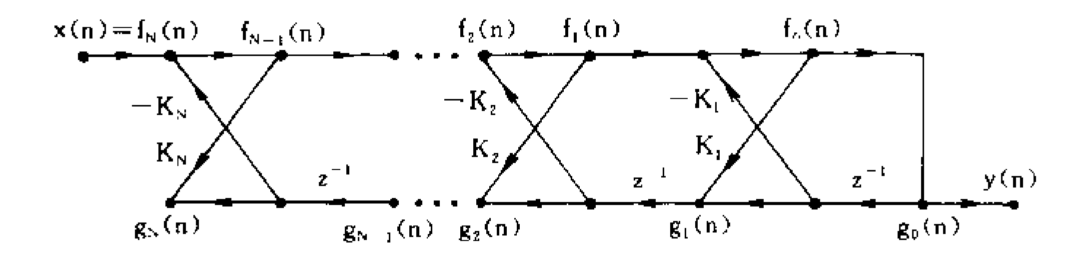


图 1.12　全极点格形滤波器

1.6　FIR 滤波器设计

1.6.1　线性相位 FIR 滤波器特性

$$H(z) = \sum_{n=0}^{M-1} h(n) z^{-n}$$

$$H(e^{j\omega}) = \sum_{n=0}^{M-1} h(n) e^{-j\omega n} \qquad \pi < \omega \leqslant \pi$$

当 FIR 滤波器具有线性相位时，即

$$\angle H(e^{j\omega}) = \beta - \alpha\omega$$

这时 h(n)具有下列一些特性：

• $\beta=0$,M 为奇数,这时 $\alpha=(M-1)/2$ 为一整数,因此 h(n)以中心点(M−1)/2 对称：

$$h(n) = h(M-1-n) \qquad 0 \leqslant n \leqslant (M-1)$$

• $\beta=0$，M 为偶数，这时 $\alpha=(M-1)/2$ 为非整数，因此 h(n)以点(M/2)−1 与点 M/2 之间中心对称：

$$h(n) = h(M-1-n) \qquad 0 \leqslant n \leqslant (M-1)$$

• $\beta=\pm\pi/2$，M 为奇数，这时 $\alpha=(M-1)/2$ 为一整数，因此 h(n)以点(M−1)/2 为中心反对称：

$$h(n) = -h(M-1-n) \qquad 0 \leqslant n \leqslant (M-1)$$

这时 $h((M-1)/2)=0$。

• $\beta=\pm\pi/2$，M 为偶数，这时 $\alpha=(M-1)/2$ 为非整数，因此 h(n)以点(M/2)−1 与点 M/2 之间中心反对称：

$$h(n) = -h(M-1-n) \qquad 0 \leqslant n \leqslant (M-1)$$

根据这 4 类 FIR 滤波器，可得到相应的频率特性的特点：

$$H(e^{j\omega}) = H_r(\omega)e^{j(\beta-\alpha\omega)}$$

其中 $H_r(\omega)$为振幅响应，它与幅值特性$|H(e^{j\omega})|$不同，β 可取 0 或$\pm\pi/2$，$\alpha=(M-1)/2$。这样可得到 4 类 FIR 滤波器。

1. 线性相位 FIR 滤波器类型 1

特点：$\beta=0$ ，对称的冲激响应，M 为奇数。

$$H(e^{j\omega}) = \left[\sum_{n=0}^{(M-1)/2} a(n)\cos\omega n\right] e^{-j\omega\frac{M-1}{2}}$$

其中

$$a(0) = h\left(\frac{M-1}{2}\right)$$

$$a(n) = 2h\left(\frac{M-1}{2} - n\right) \qquad 1 \leqslant n \leqslant \frac{M-3}{2}$$

这样就得到了

$$H_r(\omega) = \sum_{n=0}^{(M-1)/2} a(n)\cos\omega n$$

2. 线性相位 FIR 滤波器类型 2

特点：$\beta=0$，对称的冲激响应，M 为偶数。

$$H(e^{j\omega}) = \left[\sum_{n=1}^{M/2} b(n)\cos\left\{\omega\left(n-\frac{1}{2}\right)\right\}\right] e^{-j\omega\frac{M-1}{2}}$$

其中

$$b(n) = 2h\left(\frac{M}{2} - n\right) \qquad n = 1, 2, \cdots, \frac{M}{2}$$

因此

$$H_r(\omega) = \sum_{n=1}^{M/2} b(n)\cos\left\{\omega\left(n-\frac{1}{2}\right)\right\}$$

3. 线性相位 FIR 滤波器类型 3

特点：$\beta=\pi/2$，反对称的冲激响应，M 为奇数。

$$H(e^{j\omega}) = \left[\sum_{n=1}^{(M-1)/2} c(n)\ \sin\omega n\right] e^{j\left[\frac{\pi}{2}-\left(\frac{M-1}{2}\right)\omega\right]}$$

其中

$$c(n) = 2h\left(\frac{M-1}{2} - n\right) \qquad n = 1,\ 2,\ \cdots,\ \frac{M-1}{2}$$

因此得

$$H_r(\omega) = \sum_{n=1}^{(M-1)/2} c(n)\ \sin\omega n$$

4. 线性相位 FIR 滤波器类型 4

特点：$\beta=\pi/2$，反对称的冲激响应，M 为偶数。

$$H(e^{j\omega}) = \left[\sum_{n=1}^{M/2} d(n)\ \sin\left\{\omega\left(n - \frac{1}{2}\right)\right\}\right] e^{-j\omega\frac{M-1}{2}}$$

其中

$$d(n) = 2h\left(\frac{M}{2} - n\right) \qquad n = 1,\ 2,\ \cdots,\ \frac{M}{2}$$

因此得

$$H_r(\omega) = \sum_{n=1}^{M/2} d(n)\ \sin\left\{\omega\left(n - \frac{1}{2}\right)\right\}$$

在 MATLAB 中，我们着意设计出了计算这类 FIR 振幅响应的函数，分别为 hr-type1，hr-type2，hr-type3，hr-type4，通过它们可在给定 h 下计算出 H_r(振幅)，详见 3.6 的扩展函数。

1.6.2 利用窗函数设计 FIR 滤波器

对于理想的滤波器特性 $H(e^{j\omega})$，相应的 $h_d(n)$一般为无限时宽，从而是非因果的。为此实现时一般要加窗处理：

$$h(n) = h_d(n)w(n)$$

相应的

$$H(e^{j\omega}) = H_d(e^{j\omega}) \circledast W(e^{j\omega})$$

利用各种窗函数可设计出符合要求的滤波器。

1. 矩形窗

$$w(n) = \begin{cases} 1 & 0 \leqslant n \leqslant M-1 \\ 0 & \text{其它} \end{cases}$$

在 MATLAB 中，用 w=boxcar(M)可产生矩形窗 w 序列，利用 fir1 函数可实现带窗的滤波器。其它窗函数滤波也可用 fir1 实现。

2. Bartlett(巴特利特)窗

$$w(n)=\begin{cases}\dfrac{2n}{M-1} & 0\leqslant n\leqslant \dfrac{M-1}{2}\\ 2-\dfrac{2n}{M-1} & \dfrac{M-1}{2}\leqslant n\leqslant M-1\\ 0 & \text{其它}\end{cases}$$

在 MATLAB 中，用 w=bartlet(M)产生 w 序列。

3. Hanning(汉宁)窗

$$w(n)=\begin{cases}0.5\left[1-\cos\left(\dfrac{2\pi n}{M-1}\right)\right] & 0\leqslant n\leqslant M-1\\ 0 & \text{其它}\end{cases}$$

在 MATLAB 中，用 w=hanning(M)产生 w 序列。

4. Hamming(哈明)窗

$$w(n)=\begin{cases}0.54-0.46\cos\left(\dfrac{2\pi n}{M-1}\right) & 0\leqslant n\leqslant M-1\\ 0 & \text{其它}\end{cases}$$

在 MATLAB 中，用 w=hamming(M)产生 w 序列。

5. Blackman(布莱克曼)窗

$$w(n)=\begin{cases}0.42-0.5\cos\left(\dfrac{2\pi n}{M-1}\right)+0.08\cos\left(\dfrac{4\pi n}{M-1}\right) & 0\leqslant n\leqslant M-1\\ 0 & \text{其它}\end{cases}$$

在 MATLAB 中，用 w=blackman(M)产生 w 序列。

6. Kaiser(凯泽)窗

$$w(n)=\frac{I_0\left[\beta\sqrt{1-\left(1-\dfrac{2n}{M-1}\right)^2}\right]}{I_0[\beta]}\qquad 0\leqslant n\leqslant M-1$$

其中 $I_0[\cdot]$为第一类修正的零阶 Bessel(贝塞尔)函数，β是与 M 有关的参数，选择合适的β值，可产生各种过渡带宽和阻带衰减。

在 MATLAB 中，用 w=kaiser(M，beta)产生 w 序列。

对前 5 种窗函数进行比较，可得出如表 1.2 所示的结果。

表 1.2　窗函数特性

窗函数	过渡区宽度	最小阻带衰减
矩形窗	4π/M	21 dB
Bartlett	8π/M	25 dB
Hanning	8π/M	44 dB
Hamming	8π/M	53 dB
Blackman	12π/M	74 dB

在利用窗函数设计 FIR 滤波器时，还需要产生理想低通滤波器的冲激响应 $h_d(n)$，因

此在 3.6 扩展函数中给出产生 $h_d(n)$ 的函数 ideal_lp，它在给定 ω_c 和 M 时，得到理想低通的冲激响应 $h_d(n)$，有关内容详见 3.6。

1.6.3 频率取样设计技术

滤波器的 H(z)、H(k)、$H(e^{j\omega})$ 之间存在着关系

$$H(z)=\sum_{n=0}^{M-1}h(n)z^{-n}=\frac{1-z^{-M}}{M}\sum_{k=0}^{M-1}\frac{H(k)}{1-z^{-1}e^{j\frac{2\pi k}{M}}}$$

$$H(e^{j\omega})=\frac{1-e^{-j\omega M}}{M}\sum_{k=0}^{M-1}\frac{H(k)}{1-e^{-j\omega}e^{j\frac{2\pi k}{M}}}$$

$$H(k)=H(e^{j\frac{2\pi k}{M}})=\begin{cases}H(0) & k=0\\ H^*(M-k) & k=1,2,\cdots,M-1\end{cases}$$

在给定理想低通滤波器 $H_d(e^{j\omega})$ 时，首先选取长度 M，并以 $2\pi/M$ 等频率间隔对 $H_d(e^{j\omega})$ 取样得 H(k)，实际响应 $H(e^{j\omega})$ 可通过 H(k) 内插求出，h(n) 可由 ifft 函数求出，即 h(n)= IDFT[H(k)]。

具体设计时，有两种方法：第一种方法只是利用这种思想进行设计，而不管其逼近误差；第二种方法通过在过渡带增加取样点，以减少逼近误差。

1.6.4 最佳等波纹设计技术

等波纹最佳一致逼近准则是根据设计要求，导出一组条件，使整个逼近频率区域(通带或阻带)上逼近误差绝对值的最大值为最小。按这种准则设计的滤波器其通带和阻带内呈现等波纹幅度特性。因此称之为等波纹最佳逼近，也称为 Chebyshev(切比雪夫)逼近。

线性相位 FIR 滤波器的 4 种类型的幅度特性可统一表示成

$$H_r(\omega)=Q(\omega)P(\omega)$$

其中 $P(\omega)$ 是 $\cos(n\omega)$ 的线性组合：

$$P(\omega)=\sum_{n=0}^{L}\alpha(n)\cos\omega n$$

各种类型的 $P(\omega)$ 和 $Q(\omega)$ 如表 1.3 所示。

表 1.3 线性相位 FIR 滤波器 $Q(\omega)$ 和 $P(\omega)$

线性相位 FIR 滤波器	$Q(\omega)$	L	$P(\omega)$
类型 1	1	$\frac{M-1}{2}$	$\sum_{n=0}^{L}a(n)\cos\omega n$
类型 2	$\cos\frac{\omega}{2}$	$\frac{M}{2}-1$	$\sum_{n=0}^{L}\tilde{b}(n)\cos\omega n$
类型 3	$\sin\omega$	$\frac{M-3}{2}$	$\sum_{n=0}^{L}\tilde{c}(n)\cos\omega n$
类型 4	$\cos\frac{\omega}{2}$	$\frac{M}{2}-1$	$\sum_{n=0}^{L}\tilde{d}(n)\cos\omega n$

通过加窗后的滤波器误差为

$$E(\omega)\triangleq W(\omega)[H_{dr}(\omega)-H_r(\omega)]$$

$$\omega\in S\triangleq[0,\omega_p]\cup[\omega_s,\pi]$$

因此问题是确定 $a(n)$或 $\tilde{b}(n)$或 $\tilde{c}(n)$或 $\tilde{d}(n)$，使

$$\min[\max_{\omega\in S}|E(\omega)|]$$

结果得到一个以切比雪夫多项式加权的等波纹滤波器。

Parks 和 McClellan 利用 Remez 交换算法可迭代求解出滤波器系数。但一般只给定 δ_1，δ_2，ω_p 和 ω_s，而 M 未知。算法中必须提供 M 值，幸好有近似公式

$$M = \frac{-20\log_{10}\sqrt{\delta_1\delta_2}-13}{14.6\Delta f}+1$$

$$\Delta f = \frac{\omega_s-\omega_p}{2\pi}$$

当然，在实际中应适当调整 M，使阻带衰减符合设计要求。

在 MATLAB 中，这可由信号处理工具箱中的 remez 和 remezord 函数实现。

1.7 IIR 滤波器设计

IIR 滤波器具有无限宽的冲激响应，因而与模拟滤波器相匹配，而模拟滤波器的设计可借助于许多图表和公式，因此设计 IIR 滤波器有两种方法。第一种方法先设计模拟低通滤波器，然后通过频带变换而成为其它频带选择滤波器(带通、高通等)，最后通过滤波器变换而得到数字域的 IIR 滤波器。第二种方法先设计模拟低通滤波器，然后通过滤波器变换而得到数字域的低通滤波器，最后通过频带变换而得到期望 IIR 滤波器。

指定低通滤波器的性能可用频率响应的幅值平方表示(如图 1.13 所示)：

$$\frac{1}{\sqrt{1+\varepsilon^2}} \leqslant |H_a(j\Omega)|^2 \leqslant 1 \qquad |\Omega| \leqslant \Omega_p$$

$$0 \leqslant |H_a(j\Omega)|^2 \leqslant \frac{1}{A^2} \qquad \Omega_s \leqslant |\Omega|$$

这样，可得出

$$R_p = -10\log_{10}\frac{1}{1+\varepsilon^2} \qquad \varepsilon = \sqrt{10^{R_p/10}-1}$$

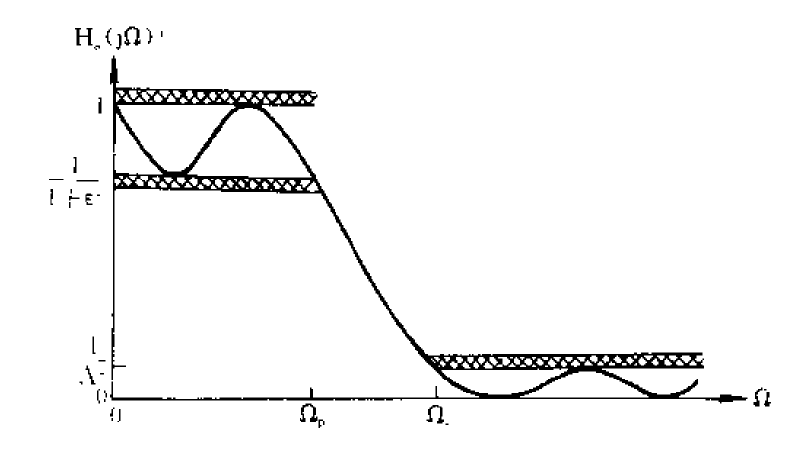


图 1.13 模拟低通滤波器指标

$$A_s = -10\log_{10}\frac{1}{A^2} \qquad A = 10^{A_s/20}$$

1.7.1 模拟滤波器原型特性

设计 IIR 滤波器的基础是设计模拟低通滤波器的原型，这些原型滤波器有：Butterworth、Chebyshev 和 Elliptic（椭圆）低通滤波器。

1. Butterworth 低通滤波器

N 阶低通 Butterworth 滤波器频域响应幅值平方为

$$|H_a(j\Omega)|^2 = \frac{1}{1+\left(\frac{\Omega}{\Omega_c}\right)^{2N}}$$

系统函数 $H_a(s)$ 可写成

$$H_a(s)H_a(-s) = |H_a(j\Omega)|^2_{\Omega=s/j} = \frac{1}{1+\left(\frac{s}{j\Omega_c}\right)^{2N}} = \frac{(j\Omega_c)^{2N}}{s^{2N}+(j\Omega_c)^{2N}}$$

在 MATLAB 中，可利用信号处理工具箱函数 buttap 函数设计归一化的（$\Omega_c=1$）N 阶 Butterworth 模拟低通滤波器原型。当 $\Omega_c\neq 1$ 时，我们设计出了扩展函数 u_buttap，同时提供了 sdir2cas 函数将其直接形式转换成级联形式，详见 3.7 的扩展函数。

在设计中，一般给定 R_p 和 A_s，但我们需知道 N 和 Ω_c，它们之间存在着关系：

（1）当 $\Omega=\Omega_p$，$-10\log_{10}|H_a(j\Omega)|^2=R_p$

即

$$-10\log_{10}\left[\frac{1}{1+\left(\frac{\Omega_p}{\Omega_c}\right)^{2N}}\right] = R_p$$

（2）在 $\Omega=\Omega_s$，$-10\log_{10}|H_a(j\Omega)|^2=A_s$

即

$$-10\log_{10}\left[\frac{1}{1+\left(\frac{\Omega_s}{\Omega_c}\right)^{2N}}\right] = A_s$$

由这两式可解出 N 和 Ω_c

$$N = \frac{\log_{10}[(10^{R_p/10}-1)(10^{A_s/10}-1)]}{2\log_{10}(\Omega_p/\Omega_s)}$$

一般 N 取比计算结果稍大的整数。

为满足 Ω_p 上的指标

$$\Omega_c = \frac{\Omega_p}{2N\sqrt{10^{R_p/10}-1}}$$

为满足 Ω_s 上的指标

$$\Omega_c = \frac{\Omega_c}{2N\sqrt{10^{A_s/10}-1}}$$

在 MATAB 中，这一设计过程可由扩展函数 afd_butt 来实现，它要求给出低通滤波器的 ω_p，ω_s，R_p 和 A_s，函数 fad_butt 可得到满足这些指标的低通 Butterworth 滤波器系数

(b, a)，详见 3.7。

2. Chebyshev 低通滤波器

有两种类型的 Chebyshev 滤波器，Chebyshev Ⅰ型滤波器在通带内具有等波纹特性，Chebyshev Ⅱ型滤波器在阻带内具有等波纹特性。

Chebyshev Ⅰ型滤波器：

$$|H_a(j\Omega)|^2 = \frac{1}{1+\varepsilon^2 T_N^2\left(\frac{\Omega}{\Omega_c}\right)}$$

其中 N 为滤波器阶，ε 为通带波纹系数(与 R_p 有关)，$T_N(x)$为 N 阶 Chebyshev 多项式

$$T_N(x) = \begin{cases} \cos(N\cos^{-1}(x)) & 0 \leqslant x \leqslant 1 \\ \cosh(\cosh^{-1}(x)) & 1 < x < \infty \end{cases}$$

在 MATLAB 中可利用信号处理工具箱函数 Cheb1ap 函数设计归一化的 Chebyshev Ⅰ型模拟原型滤波器。当 $\Omega_c \neq 1$ 时，我们给出了扩展函数 u_chb1ap 来进行设计，有关内容详见 3.7。

在具体设计时，给定 Ω_p，Ω_s，R_p 及 A_s，要确定 Chebyshev Ⅰ型滤波器的 ε，Ω_c，N，它们之间的关系为

$$\varepsilon = \sqrt{10^{0.1R_p}-1}$$

$$A = 10^{A_s/20}$$

$$\Omega_c = \Omega_p$$

$$\Omega_r = \frac{\Omega_s}{\Omega_p}$$

$$g = \sqrt{(A^2-1)/\varepsilon^2}$$

$$N = \frac{\log_{10}[g+\sqrt{g^2-1}]}{\log_{10}[\Omega_r+\sqrt{\Omega_r^2-1}]}$$

其中 N 应取稍大的整数。

这一设计过程可构成 MATLAB 的扩展函数 afd_chb1，其调用格式为[b, a]=afd_chb1(Wp, Ws, Rp, As)，有关内容详见 3.7。

Chebyshev Ⅱ型滤波器：

$$|H_a(j\Omega)|^2 = \frac{V}{1+[\varepsilon T_N^2(\Omega_c/\Omega)]^{-1}}$$

在 MATLAB 中，可利用 cheb2ap 函数进行设计归一化的 Chebyshev Ⅱ型模拟滤波器。对非归一化的滤波器，可采用扩展函数 u_chb2ap 进行设计，其调用格式为[b, a]=u_chb2ap(N, As, Omegac)。由 ω_p，ω_s，R_p，A_s 可确定 N，Ω_c，从而得出滤波器设计函数 fda_chb2，其调用格式为[b, a]=afd_chb2(Wp, Ws, Rp, As)，有关内容详见 3.7。

3. 椭圆低通滤波器

$$|H_a(j\Omega)|^2 = \frac{1}{1+\varepsilon^2 U_N^2\left(\frac{\Omega}{\Omega_c}\right)}$$

其中 $U_N(\cdot)$为 N 阶雅可比椭圆函数。阶数 N 由下式确定

$$N=\frac{K(k)\,K(\sqrt{1-k_1^2})}{K(k_1)\,K(\sqrt{1-k^2})}$$

其中

$$k=\frac{\Omega_p}{\Omega_s}\qquad k_1=\frac{\varepsilon}{\sqrt{A^2-1}}$$

$$K(x)=\int_0^{\pi/2}\frac{d\theta}{\sqrt{1-x^2\sin^2\theta}}$$

K(x)为第一类椭圆积分，MATLAB 专门为此提供了计算函数 ellipke。

MATLAB 为归一化椭圆模拟原型滤波器的设计提供了函数 ellipap，对非归一化椭圆模拟原型滤波器的设计，给出了扩展函数 u_elipap，其调用格式为[b，a]=u_elipap(N，Rp，As，Omegac)。利用 u_elipap 函数可设计出设计模拟椭圆低通滤波器的扩展函数 afd_elip，其调用格式为[b，a]=afd_elip(Wp，Ws，Rp，As)，有关内容详见 3.7。

1.7.2 模拟到数字滤波器变换

从模拟滤波器到数字滤波器可采用各种变换技术，最重要的有两种：冲激不变法(保持冲激响应不变)和双线性变换(保持系统函数表示不变)。

1. 冲激不变变换

设取样时间为 T，则冲激不变为

$$h(n)=h_a(nT)$$

模拟频率 Ω 与数字频率 ω 之间存在的关系为

$$\omega=\Omega T\quad e^{j\omega}=e^{j\Omega T}$$

s 平面与 z 平面之间有

$$z=e^{sT}$$

系统函数之间有

$$H(z)=\frac{1}{T}\sum_{k=-\infty}^{\infty}H_a\left(s-j\frac{2\pi}{T}k\right)$$

在给定数字低通滤波器指标 ω_p，ω_s，R_p 和 A_s，设计 H(z)可分以下 4 步：

(1) 选 T，确定模拟频率

$$\Omega_p=\frac{\omega_p}{T}\qquad\Omega_s=\frac{\omega_s}{T}$$

(2) 利用指标 Ω_p，Ω_s，R_p 和 A_s 设计模拟滤波器 $H_a(s)$(这是前一小节的内容)。

(3) 利用部分分式展开，将 $H_a(s)$写成

$$H_a(s)=\sum_{k=1}^{N}\frac{R_k}{s-P_k}$$

(4) 将模拟极点 P_k 变换成数字极点$\{e^{P_kT}\}$，这样得

$$H(z)=\sum_{k=1}^{N}\frac{R_k}{1-e^{P_kT}z^{-1}}$$

这一过程可设计成 MATLAB 的扩展函数 imp_invr，其调用格式为[b，a]=imp_invr(c，d，T)，其中 $H_a(s)=c(s)/d(s)$，$H(z)=b(z)/a(z)$，T 为取样时间，有关内容详见 3.7。

2. 双线性变换

双线性变换是 s 域与 z 域之间的最佳映射：

$$s = \frac{2}{T}\frac{1 - z^{-1}}{1 + z^{-1}} \qquad z = \frac{1 + sT/2}{1 - sT/2}$$

或写成

$$\frac{T}{2}sz + \frac{T}{2}s - z + 1 = 0$$

因此 s 与 z 之间为线性关系。

利用 $s=\sigma+j\Omega$ 得

$$z = \frac{1 + \frac{\sigma T}{2} + j\frac{\Omega T}{2}}{1 - \frac{\sigma T}{2} - j\frac{\Omega T}{2}}$$

由 $\sigma=0$ 得

$$z = \frac{1 + j\frac{\Omega}{2}}{1 - j\frac{\Omega}{2}}$$

因此

$$\omega = 2\tan^{-1}\left(\frac{\Omega T}{2}\right) \quad \Omega = \frac{2}{T}\tan\left(\frac{\omega}{2}\right)$$

当给定滤波器指标 ω_p，ω_s，R_p，A_s 时，其设计步骤如下：

(1) 选 T，通常选 T=1。

(2) 对 ω_p 和 ω_s 作预畸变

$$\Omega_p = \frac{2}{T}\tan\left(\frac{\omega_p}{2}\right) \quad \Omega_s = \frac{2}{T}\tan\left(\frac{\omega_s}{2}\right)$$

(3) 按指标 Ω_p，Ω_s，R_p，A_s 设计模拟滤波器。

(4) 最后，由

$$H(z) = H_a\left(\frac{2}{T}\frac{1 - z^{-1}}{1 + z^{-1}}\right)$$

求出 H(z)，这可由 MATLAB 的 bilinear 函数实现。

1.7.3 低通滤波器设计

低通滤波器的设计可直接采用信号处理工具箱提供的 M 函数：butter，cheby1，cheby2 及 ellip。它们要求提供滤波器阶 N 及截止频率 ω_n 等参数，为此，又给出确定 N 和 ω_n 的函数 buttord，cheb1ord，cheb2ord 和 ellipord。这些函数的用法可参见第 2 章内容。

1.7.4 频带变换

在设计出低通滤波器后，可通过频带变换得到带通、高通和带阻滤波器。设低通滤波器为 $H_{LP}(z)$，期望得到的频选滤波器为 H(z)，则可定义一映射

$$Z^{-1} = G(z^{-1})$$

使得

$$H(z) = H_{LP}(Z)|_{Z^{-1} \leftarrow G(z^{-1})}$$

这种映射关系应满足一定的条件才能从低通滤波器得到期望的频选滤波器，如表 1.4 所示。

表 1.4　数字滤波器的频率变换(低通滤波器原型的截止频率为 ω_c')

变换类型	变　换	参　数
低通	$z^{-1} \to \frac{z^{-1}-\alpha}{1-\alpha z^{-1}}$	$\alpha = \frac{\sin\left(\frac{\omega_c'-\omega_c}{2}\right)}{\sin\left(\frac{\omega_c'+\omega_c}{2}\right)}$
高通	$z^{-1} \to -\frac{z^{-1}+\alpha}{1+\alpha z^{-1}}$	$\alpha = -\frac{\cos\left(\frac{\omega_c'+\omega_c}{2}\right)}{\cos\left(\frac{\omega_c'-\omega_c}{2}\right)}$
带通	$z^{-1} \to -\frac{z^{-2}-\alpha_1 z^{-1}+\alpha_2}{\alpha_2 z^{-2}-\alpha_1 z^{-1}+1}$	$\alpha_1 = -\frac{2\beta K}{K+1}$ $\alpha_2 = \frac{K-1}{K+1}$ $\beta = \frac{\cos\left(\frac{\omega_u+\omega_l}{2}\right)}{\cos\left(\frac{\omega_u-\omega_l}{2}\right)}$ $K = \cot\frac{\omega_u-\omega_l}{2}\tan\frac{\omega_c'}{2}$
带阻	$z^{-1} \to \frac{z^{-2}-\alpha_1 z^{-1}+\alpha_2}{\alpha_2 z^{-2}-\alpha_1 z^{-1}+1}$	$\alpha_1 = -\frac{2\beta}{K+1}$ $\alpha_2 = \frac{K-1}{K+1}$ $\beta = \frac{\cos\left(\frac{\omega_u+\omega_l}{2}\right)}{\cos\left(\frac{\omega_u-\omega_l}{2}\right)}$ $K = \tan\frac{\omega_u-\omega_l}{2}\tan\frac{\omega_c'}{2}$

将表 1.4 中的映射关系代入 $H_{LP}(Z)$，从而得到 $H(z)$，这种过程可借助于 conv 函数来计算。这里给出一个扩展函数 zmapping，其调用格式为[bz, az]=zmazpping(bZ,aZ,Nz,Dz)，其中 $H_{LP}(z)=bZ/aZ$，$z^{-1}=G(z^{-1})=N(z)/D(z)$，结果得到 $H(z)=bZ/aZ$，有关内容详见 3.7。

在具体设计过程中，应确定低通滤波器的截止频率。从理论上说，应该按期望的频选特性及 α 值确定 ω_c'；但实际中，可选定 ω_c'(如 $\omega_c'=0.2\pi$)，而后根据表 1.4 确定 α 值，从而设计出期望的频带滤波器。这一过程也可写成 MATLAB 的扩展函数，例如，为设计高通型 Chebysh Ⅰ型数字滤波器，可设计出 cheb1hpf 函数，其调用格式为[b, a]=cheb1hpf(wp, ws, Rp, As)，详见 3.7。

第2章

信号处理工具箱函数

MATLAB 包含了进行信号处理的许多工具箱函数，有关这些工具箱函数的使用可通过 Help 命令得到。为使用方便，本章将给出这些函数的使用说明。为方便用户查询，本节先简要地分组列出各种工具箱函数(如表 2.1～2.15 所示)，然后按字母顺序排列给出工具箱函数的索引(如表 2.16 所示)。

表 2.1 波形产生

函数名	功　　能
sawtooth	产生锯齿波或三角波
square	产生方波
sinc	产生 sinc 或$\frac{\sin(\pi t)}{\pi t}$函数
diric	产生 Dirichlet 或周期 sinc 函数

表 2.2 滤波器分析和实现

函数名	功　　能
abs	求绝对值(幅值)
angle	求相角
conv	求卷积
fftfilt	重叠相加法 FFT 滤波器实现
filter	直接滤波器实现
filtfilt	零相位数字滤波
filtic	filter 函数初始条件选择
freqs	模拟滤波器频率响应

续表

函数名	功　　能
freqspace	频率响应中的频率间隔
freqz	数字滤波器频率响应
grpdelay	平均滤波器延迟(群延迟)
impz	数字滤波器的冲激响应
zplane	离散系统零极点图

表 2.3　线性系统变换

函数名	功　　能
convmtx	卷积矩阵
poly2rc	从多项式系数中计算反射系数
rc2poly	从反射系数中计算多项式系数
residuez	Z 变换部分分式展开或留数计算
sos2ss	变系统二阶分割形式为状态空间形式
sos2tf	变系统二阶分割形式为传递函数形式
sos2zp	变系统二阶分割形式为零极点增益形式
ss2sos	变系统状态空间形式为二阶分割形式
ss2tf	变系统状态空间形式为传递函数形式
ss2zp	变系统状态空间形式为零极点增益形式
tf2ss	变系统传递函数形式为状态空间形式
tf2zp	变系统传递函数形式为零极点增益形式
zp2sos	变系统零极点增益形式为二阶分割形式
zp2ss	变系统零极点增益形式为状态空间形式
zp2tf	变系统零极点增益形式为传递函数形式

表 2.4　IIR 滤波器设计

函数名	功　　能
besself	Bessel(贝塞尔)模拟滤波器设计
butter	Butterworth(比特沃思)滤波器设计
cheby1	Chebyshev(切比雪夫)Ⅰ型滤波器设计
cheby2	Chebyshev(切比雪夫)Ⅱ型滤波器设计
ellip	椭圆滤波器设计
yulewalk	递归数字滤波器设计

表 2.5 IIR 滤波器阶的选择

函数名	功　　能
buttord	Butterworth 滤波器阶的选择
cheb1ord	Chebyshev Ⅰ 型滤波器阶的选择
cheb2ord	Chebyshev Ⅱ 型滤波器阶的选择
ellipord	椭圆滤波器阶的选择

表 2.6 FIR 滤波器设计

函数名	功　　能
fir1	基于窗函数的 FIR 滤波器设计——标准响应
fir2	基于窗函数的 FIR 滤波器设计——任意响应
firls	最小二乘 FIR 滤波器设计
intfilt	内插 FIR 滤波器设计
remez	Parks-McCellan 最优 FIR 滤波器设计
remezord	Parks-McCellan 最优 FIR 滤波器阶估计

表 2.7 变　　换

函数名	功　　能
czt	线性调频 Z 变换
dct	离散余弦变换(DCT)
idct	逆离散余弦变换
dftmtx	离散傅里叶变换矩阵
fft	一维快速傅里叶变换
ifft	一维逆快速傅里叶变换
fftshift	重新排列 FFT 的输出
hilbert	Hilbert(希尔伯特)变换

表 2.8 统计信号处理

函数名	功　　能
cov	协方差矩阵
xcov	互协方差函数估计
corrcoef	相关系数矩阵
xcorr	互相关函数估计
cohere	相关函数平方幅值估计
csd	互谱密度(CSD)估计
psd	信号功率谱密度(PSD)估计
tfe	从输入输出中估计传递函数

表 2.9 窗 函 数

函数名	功 能
boxcar	矩形窗
triang	三角窗
bartlett	Bartlett(巴特利特)窗
hamming	Hamming(哈明)窗
hanning	Hanning(汉宁)窗
blackman	Blackman(布莱克曼)窗
chebwin	Chebyshev(切比雪夫)窗
kaiser	Kaiser(凯泽)窗

表 2.10 参数化建模

函数名	功 能
invfreqs	模拟滤波器拟合频率响应
invfreqz	离散滤波器拟合频率响应
prony	利用 Prony 法的离散滤波器拟合时间响应
stmcb	利用 Steiglitz-McBride 迭代方法求线性模型
levinson	Levinson-Durbin 递归算法
lpc	线性预测系数

表 2.11 特殊操作

函数名	功 能
rceps	实倒谱和最小相位重构
cceps	倒谱分析和最小相位重构
decimate	降低序列的取样速率
interp	提高取样速率(内插)
resample	改变取样速率
medifilt1	一维中值滤波
deconv	反卷积和多项式除法
modulate	通讯仿真中的调制
demod	通讯仿真中的解调
vco	电压控制振荡器
specgram	频谱分析

表 2.12 模拟原型滤波器设计

函数名	功　　能
besselap	Bessel 模拟低通滤波器原型
buttap	Butterworth 模拟低通滤波器原型
cheb1ap	Chebyshev Ⅰ 型模拟低通滤波器原型
cheb2ap	Chebyshev Ⅱ 型模拟低通滤波器原型
ellipap	椭圆模拟低通滤波器原型

表 2.13 频率变换

函数名	功　　能
lp2bp	低通到带通模拟滤波器变换
lp2hp	低通到高通模拟滤波器变换
lp2bs	低通到带阻模拟滤波器变换
lp2lp	低通到低通模拟滤波器变换

表 2.14 滤波器离散化

函数名	功　　能
bilinear	双线性变换
impinvar	冲激响应不变法实现模拟到数字的滤波器变换

表 2.15 其　它

函数名	功　　能
conv2	二维卷积
cplxpair	将复数归成复共轭对
detrend	删除线性趋势
fft2	二维快速傅里叶变换(FFT)
ifft2	二维逆 FFT 变换
filter2	二维数字滤波器
polystab	稳定多项式
strips	带状图
xcorr2	二维互相关系数

表 2.16 函数索引表

函数名	页码	函数名	页码	函数名	页码
abs	46	ellipord	80	poly2rc	56
angle	46	fft	91	polystab	125
bartlett	102	fft2	124	prony	106
besselap	117	fftfilt	47	psd	99
besself	64	fftshift	93	rc2poly	57
bilinear	121	filter	47	rceps	109
blackman	103	filter2	125	remez	86
boxcar	101	filtfilt	48	remezord	88
buttap	118	filtic	48	resample	112
butter	66	fir1	82	residuez	57
buttord	75	fir2	83	sawtooth	43
cceps	110	firls	85	sinc	44
cheb1ap	118	freqs	49	sos2ss	58
cheb1ord	77	freqspace	49	sos2tf	59
cheb2ap	118	freqz	51	sos2zp	59
cheb2ord	79	grpdelay	52	specgram	116
chebwin	103	hamming	102	square	44
cheby1	68	hanning	103	ss2sos	60
cheby2	70	hilbert	93	ss2tf	61
cohere	96	idct	90	ss2zp	61
conv	46	ifft	92	stmcb	107
conv2	123	ifft2	124	strips	125
convmtx	55	impinvar	122	tf2ss	61
corrcoef	95	impz	53	tf2zp	62
cov	94	interp	111	tfe	100
cplxpair	123	intfilt	86	triang	102
csd	98	invfreqs	104	vco	115
czt	89	invfreqz	105	xcorr	95
dct	90	kaiser	104	xcorr2	126
decimate	110	levinson	108	xcov	94
deconv	113	lp2bp	119	yulewalk	73
demod	115	lp2bs	120	zp2sos	63
detrend	124	lp2hp	120	zp2ss	64
dftmtx	91	lp2lp	121	zp2tf	64
diric	44	lpc	108	zplane	54
ellip	72	medfilt1	113		
ellipap	119	modulate	113		

最后，我们列出本章用到的符号及其含义，如表 2.17 所示。

本章的 2.1 节～2.15 节分类列出信号处理工具箱函数的简要说明，包括函数名、功能、使用格式、说明及与此相关的函数。

表 2.17 函数说明中的符号

符号	含 义	符号	含 义
ai	多项式系数初值	num	分子多项式
alpha	系数 α	opt	可选参数
den	分母多项式	order	顺序格式
duty	工作周期	Rp	通带波纹
Fc	载波频率	Rs	阻带波纹
Fs	采样频率	sd	时宽
ftype	滤波器类型	tol	误差容限
h	幅值	w	权值或频率
iter	迭代次数	width	宽度
iu	序号	window	窗函数
lap	区域大小	Wn	频率
nfft	FFT 的长度	Wp	通带截止频率
noverlap	覆盖点数	Ws	阻带截止频率
npt	点数	Wt	加权矢量

2.1 波 形 产 生

1. sawtooth

功能：产生锯齿波或三角波。

格式：

x＝sawtooth(t)

x＝sawtooth(t，width)

说明：

sawtooth(t)函数类似于 sin(t)，它产生周期为 2π，幅值从－1 到＋1 的锯齿波，在 2π 的整数倍处，其值为－1，并以 $1/\pi$ 的斜率线性上升至＋1。

sawtooth(t，width)用于产生三角波，其 width(0＜width≤1 的标量)用于确定最大值的位置，即从 0 到 $2\pi*$width 函数从－1 上升到＋1，然后在 $2\pi*$width 至 2π 之间又线性地从＋1 降至－1，周而复始。例如，当 width＝0.5 时，可产生一对称的标准三角波，当 width＝1 时，就产生锯齿波，即 sawtooth(t，1)＝sawtooth(t)。

参见：square

2. square

功能：产生方波。

格式：

x=square(t)

x=square(t, duty)

说明：

square(t)类似于 sin(t)，产生周期是 2π，幅值为±1 的方波，square(t, duty)产生指定周期的方波，其 duty 用于指定正半周期的比例。

参见：sawtooth

3. sinc

功能：产生 sinc 或$\frac{\sin(\pi t)}{\pi t}$函数。

格式：

y=sinc(x)

说明：

MATLAB 中的函数 sinc 可用于计算 sinc 函数，即

$$\text{sinc}(t) = \begin{cases} 1 & t = 0 \\ \dfrac{\sin(\pi t)}{\pi t} & t \neq 0 \end{cases}$$

这个函数正好是宽度为 2π，幅度为 1 的矩形脉冲的连续逆傅里叶变换，即

$$\text{sinc}(t) = \frac{1}{2\pi}\int_{-\pi}^{\pi} e^{j\omega t} d\omega$$

参见：diric, sawtooth, square

4. diric

功能：产生 Dirichlet 或周期 sinc 函数。

格式：

y=diric(x, n)

说明：

在 y=diric(x, n)中，n 必须为正整数，y 为相应的 x 元素的 Dirichlet 函数，即

$$\text{dirichlet}(x) = \begin{cases} (-1)^{k(n-1)} & x = 2\pi k,\ k = 0,\ \pm 1,\ \pm 2,\ \cdots \\ \dfrac{\sin(nx/2)}{n\sin(x/2)} & \text{其它} \end{cases}$$

dirichlet 函数是周期信号，当 n 为奇数时，周期为 2π；当 n 为偶数时，周期为 4π。

例 下面这段程序可分别得到 sinc(x)和 diric(x, n)的曲线，如图 2.1 和图 2.2 所示，从中可看出 diric 与 sinc 函数的区别。

```
x=[-4*pi:0.1/pi:4*pi];
```

```
y=sinc(x);
figure(1)
plot(x, y, 'w')
title('sinc function y=sinc(x)')
xlable('x')
ylabel('y')
figure(2)
y1=diric(x, 7);
plot(x, y1, 'w')
title('Dirichlet function y=diric(x, n) n=7')
xlabel('x')
ylabel('y')
```

图 2.1 sinc 函数

图 2.2 Dirichlet(周期 sinc)函数

2.2 滤波器分析和实现

1. abs

功能：求绝对值(幅值)。

格式：

y=abs(x)

说明：

y=abs(x)用于计算 x 的绝对值。当 x 为复数时，得到的是复数模(幅值)，即

$$abs(x)=\sqrt{(Re(x))^2+(Im(x))^2}$$

当 x 为字符串时，abs(x)得到字符串的各个字符的 ASCII 码，例如 x='123'，则 abs(x)得到 49 50 51。

参见：angle

2. angle

功能：求相角。

格式：

p=angle(h)

说明：

p=angle(h)用于求取复矢量或复矩阵 h 的相角(以弧度为单位)，相角介于 $-\pi$ 和 $+\pi$ 之间。例如对某复数 h 可用两种方法表示

$$h=x+i\,y=m\,e^{ip}$$

则幅值 m 和相角 p 可由 x+i y 格式求出

m=abs(h)

p=angle(h)

当然，由 m 和 p 也可求取 x+i y 格式

h=m. * exp(i * p)

x=real(h)

y=imag(h)

参见：abs

3. conv

功能：求卷积。

格式：

c=conv(a, b)

说明：

conv(a, b)用于求取矢量 a 和 b 的卷积，即

$$c(n+1)=\sum_{k=0}^{N-1}a(k+1)b(n-k)$$

其中 N 为矢量 a 和 b 的最大长度。例如，当 a=[1 2 3]，b=[4 5 6]时，则

```
c=conv(a, b)
c=
    4 13 28 27 18
```

参见：conv2，convmtx，deconv，filter，residuez

4. fftfilt

功能：利用重叠相加法的基于 FFT 的 FIR 滤波。

格式：

```
y=fftfilt(b, x)
y=fftfilt(b, x, n)
```

说明：

fftfilt 利用基于 FFT 的重叠相加法对数据进行滤波，这种频域滤波技术只对 FIR 滤波器有效。

y=fftfilt(b, x)通过系数矢量 b 描述的滤波器对矢量 x 中的数据进行滤波。在时域中滤波可由差分方程表示

$$y(n)=b(1)x(n)+b(2)x(n-1)+\cdots+b(n_b+1)x(n-n_b)$$

也可在 z 域或频域内表示

$$Y(z)=(b(1)+b(2)z^{-1}+\cdots+b(n_b+1)z^{-n_b})X(z)$$

在缺省情况下，fftfilt 会选择 FFT 的长度及数据块长度，以保证合适的执行时间。

y=fftfilt(b, x, n)采用 FFT 的长度为 $n_{fft}=2$ ^ nextpow2(n)，数据块长度取为 $n_{fft}-$ length(b)+1。其中 nextpow2(n)函数可找出大于 n 而与 n 最接近的 2 的幂指数，例如，当 n=33 时，nextpow2(n)=6，而当 n=32 时，nextpow2(n)=5。

矢量 x 既可以为实数，也可以为复数。

参见：filter，filtfilt

5. filter

功能：利用递归滤波器(IIR)或非递归滤波器(FIR)对数据进行滤波。

格式：

```
y=filter(b, a, x)
[y, zf]=filter(b, a, x)
y=filter(b, a, x, zi)
```

说明：

filter 采用数字滤波器对数据进行滤波，其实现采用移位直接型 Ⅱ 结构，因而适用于

IIR 和 FIR 滤波器。滤波器的 z 域表示为

$$Y(z)=\frac{b(1)+b(2)z^{-1}+\cdots+b(n_b+1)z^{-n_b}}{1+a(2)z^{-1}+\cdots+a(n_a+1)z^{-n_a}}X(z)$$

这里 a(1)=1，如果输入的 a(1)≠1，则 MATLAB 将自动进行归一化系数的操作；如果 a(1)=0，则给出出错信息。

y=filter(b，a，x)利用给定的矢量 a 和 b，对 x 中的数据进行滤波，结果放入 y 矢量，其长度取 $\max(n_a, n_b)$。

[y，zf]=filter(b，a，x)除得到结果矢量 y 外，还得到 x 的最终状态矢量 zf。

y=filter(b，a，x，zi)可在 zi 中指定 x 的初始状态。

参见：fftfilt，filter2，filtfilt，filtic

6. filtfilt

功能：零相位数字滤波。

格式：

y=filtfilt(b，a，x)

说明：

y=filtfilt(b，a，x)通过将输入数据前向和反向处理，以完成零相位数字滤波。它先将数据按顺序滤波，然后将所得结果逆转后反向通过滤波器，这样得到的序列为精确零相位失真，并使滤波器的阶加倍。filtfilt 通过与初始条件相匹配，可使起始和结束阶段的暂态过渡过程最短。

filtfilt 要调用 filter 函数，因此参数矢量 a 和 b 的含义可参见 filter。

参见：fftfilt，filter，filter2

7. filtic

功能：为 filter 函数选择初始条件。

格式：

z=filtic(b，a，y，x)

z=filtic(b，a，y)

说明：

filtic 函数在给定过去输出 y 和输入 x 的前提下，为移位直接形式Ⅱ滤波器实现中的延迟找出初始条件 z。矢量 x 和 y 分别表示过去的输入和输出

$$x=\{x(-1), x(-2), \cdots, x(-n_b), \cdots\}$$

$$y=\{y(-1), y(-2), \cdots, y(-n_a), \cdots\}$$

其中 n_b=length(b)−1(分子阶数)，n_a=length(a)−1(分母阶数)。矢量 x 和 y 的长度应分别等同于 n_b 和 n_a，如果长度不够，则补 0；如果长度超出，则截断处理，使 x 和 y 的长度正好为 n_b 和 n_a。输出矢量 z 的长度为 $\max(n_a, n_b)$。

z=filtic(b，a，y，x)可得到给定输入 x 和输出 y 时的初始状态；z=filtic(b，a，y)假定输入 x=0。

参见：filter，filtfilt

8．freqs

功能：模拟滤波器的频率响应。

格式：

```
h=freqs(b, a, w)
[h, w]=freqs(b, a)
[h, w]=freqs(b, a, n)
freqs(b, a)
```

说明：

freqs 用于计算由矢量 a 和 b 构成的模拟滤波器

$$H(s)=\frac{B(s)}{A(s)}=\frac{b(1)s^{n_b}+b(2)s^{(n_b-1)}+\cdots+b(n_b+1)}{s^{n_a}+a(2)s^{(n_a-1)}+\cdots+a(n_a+1)}$$

的复频响应 $H(j\omega)$。

h=freqs(b，a，w)用于计算模拟滤波器的复频响应，其中实矢量 w 用于指定频率值，即 freqs 沿虚轴计算频率响应。

[h，w]=freqs(b，a)自动设定 200 个频率点来计算频率响应，这 200 个频率值记录在 w 中。

[h，w]=freqs(b，a，n)设定 n 个频率点计算频率响应。

不带输出变量的 freqs 函数，将在当前图形窗口中绘制出幅频和相频曲线。

例如，有一模拟滤波器，其传递函数设为

$$H(s)=\frac{0.2s^2+0.3s+1}{s^2+0.4s+1}$$

现可通过 freqs 函数绘制出它的幅频特性和相频特性，其程序为

```
a=[1 0.4 1];
b=[0.2 0.3 1];
w=logspace(-1, 1);
freqs(b, a, w)
```

结果如图 2.3 所示。

参见：abs，angle，freqz，invfreqs

9．freqspace

功能：频率响应中的频率间隔。

格式：

```
f=freqspace(n)
f=freqspace(n, 'whole')
[f1, f2]=freqspace(n)
[f1, f2]=freqspace([m n])
[x1, y1]=freqspace(n, 'meshgrid')
```

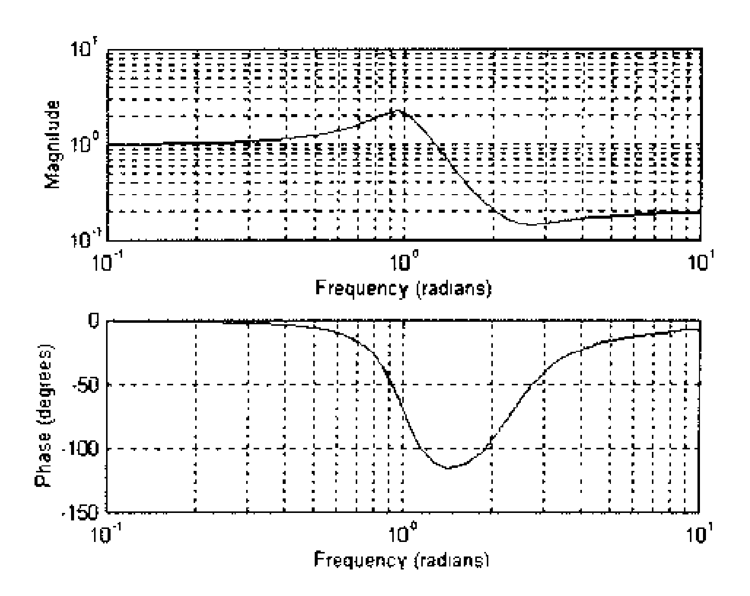


图 2.3　模拟滤波器的幅频和相频特性曲线

[x1, y1]=freqspace([m n], 'meshgrid')

说明：

freqspace 函数可产生等间隔频率响应所包含的频率范围，这在采用 freqz 函数时是非常有用的。

f=freqspace(n)可产生从 0 到 1 的均匀分布的点，即相当于 f=[0:2/n:1]，如 freqspace(5)可产生[0 0.4 0.8 1]，而 freqspace(6)得[0 0.3333 0.6667 1]。因此，针对 n 的奇偶，其点数分别为(n+1)/2 和(n+2)/2。

f=freqspace(n, 'whole')将产生 n 个均匀的点，即有 f=[0:2/n:2*(n−1)/n]，如 freqspace(5, 'whole')可得到[0 0.4 0.8 1.2 1.6]。

[f1, f2]=freqspace(n)将产生 n×n 的二维频率矢量 f1 和 f2，例如[f1, f2]=freqspace(5)，可得

```
f1=
    -0.8  -0.4  0  0.4  0.8
f2=
    -0.8  -0.4  0  0.4  0.8
```

显然计算公式为

$$f1=f2=\begin{cases}[-1+1/n:2/n:1-1/n] & n\text{ 为奇数}\\ [-1:2/n:1-2/n] & n\text{ 为偶数}\end{cases}$$

[f1, f2]=freqspace([m n])将产生 m×n 的二维矢量 f1 和 f2，例如[f1, f2]=freqspace([5 4])，可得

```
f1=
    -1.0  -0.5   0  0.5
f2=
    -0.8  -0.4   0  0.4  0.8
```

最后两种格式可等效表示

[x1, y1]=freqspace(n, 'meshgrid');

等效于

[f1, f2]=freqspace(n);

[x1, y1]=meshgrid(f1, f2);

同样,

[x1, y1]=freqspace([m n], 'meshgrid');

等效于

[f1, f2]=freqspace([m n]);

[x1, y1]=meshgrid(f1, f2);

参见: freqz, invfreqz

10. freqz

功能: 数字滤波器的频率响应。

格式:

[h, w]=freqz(b, a, n)

[h, f]=freqz(b, a, n, Fs)

[h, w]=freqz(b, a, n, 'whole')

[h, f]=freqz(b, a, n, 'whole', Fs)

h=freqz(b, a, w)

h=freqz(b, a, f, Fs)

freqz(b, a)

说明:

freqz 用于计算由矢量 a 和 b 构成的数字滤波器

$$H(z)=\frac{B(z)}{A(z)}=\frac{b(1)+b(2)z^{-1}+\cdots+b(n_b+1)z^{-n_b}}{1+a(2)z^{-1}+\cdots+a(n_a+1)z^{-n_a}}$$

的复频响应 H(jω)。

[h, w]=freqz(b, a, n)可得到数字滤波器的 n 点的复频响应，这 n 个点均匀地分布在上半单位圆(即 0～π)，并将这 n 点频率记录在 w 中，相应的频率响应记录在 h 中。至于 n 值的选择没有太多的限制，只要 n>0 的整数，但最好能选取 2 的幂次方，这样就可采用 FFT 算法进行快速计算。如果缺省，则 n=512。

[h, f]=freqz(b, a, n, Fs)允许指定采样终止频率 Fs(以 Hz 为单位)，也即在 0～Fs/2 频率范围内选取 n 个频率点(记录在 f 中)，并计算相应的频率响应 h。

[h, w]=freqz(b, a, n, 'whole')表示在 0～2π 之间均匀选取 n 个点计算频率响应。[h, f]=freqz(b, a, n, 'whole', Fs)则在 0～Fs 之间均匀选取 n 个点计算频率响应。

h=freqz(b, a, w)计算在矢量 w 中指定的频率处的频率响应，但必须注意，指定的频率必须介于 0 和 2π 之间。

h=freqz(b, a, f, Fs)计算在矢量 f 中指定的频率处的频率响应，但指定频率必须介于 0 和 Fs 之间。

不带输出变量的 freqz 函数可在当前图形窗口中绘制出幅频和相频特性曲线。

例 对一数字滤波器

$$H(z)=\frac{0.2+0.3z^{-1}+z^{-2}}{1+0.4z^{-1}+z^{-2}}$$

则可通过程序

```
b=[0.2  0.3  1];
a=[1  0.4  1];
freqz(b, a, 128)
```

得到滤波器的幅频和相频特性曲线，如图 2.4 所示。

参见：abs，angle，fft，filter，freqs，impz，invfreqz

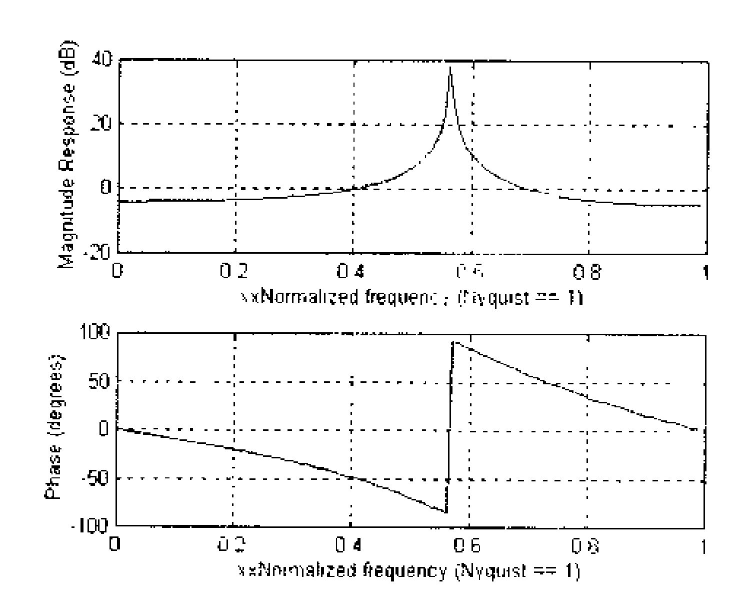


图 2.4 数字滤波器的幅频和相频特性曲线

11. grpdelay

功能：平均滤波延迟(群延迟)。

格式：

```
[gd, w]=grpdelay(b, a, n)
[gd, f]=grpdelay(b, a, n, Fs)
[gd, w]=grpdelay(b, a, n, 'whole')
[gd, f]=grpdelay(b, a, n, 'whole', Fs)
gd=grpdelay(b, a, w)
gd=grpdelay(b, a, f, Fs)
grpdelay(b, a)
```

说明：

滤波器的群延迟是滤波器平均延迟相对于频率的函数，实际上它就是滤波器相位响应的负一阶导数，即如果滤波器的频率响应为 $H(j\omega)$，其相角为 $\theta(\omega)$，则群延迟为

$$\tau_g(\omega) = -\frac{d\theta(\omega)}{d\omega}$$

grpdelay 函数与 freqz 函数相似，只是 freqz 用于计算频率响应，而 grpdelay 用于计算群延迟，其它说明可参见 freqz 函数。

例 通过最后一种格式绘制出群延迟特性。如输入程序

```
b=[0.2  0.3  1];
a=[1  0.4  1];
grpdelay(b, a, 128)
```

可得到如图 2.5 所示的滤波器群延迟特性。

参见：cceps，fft，freqz，hilbert，receps

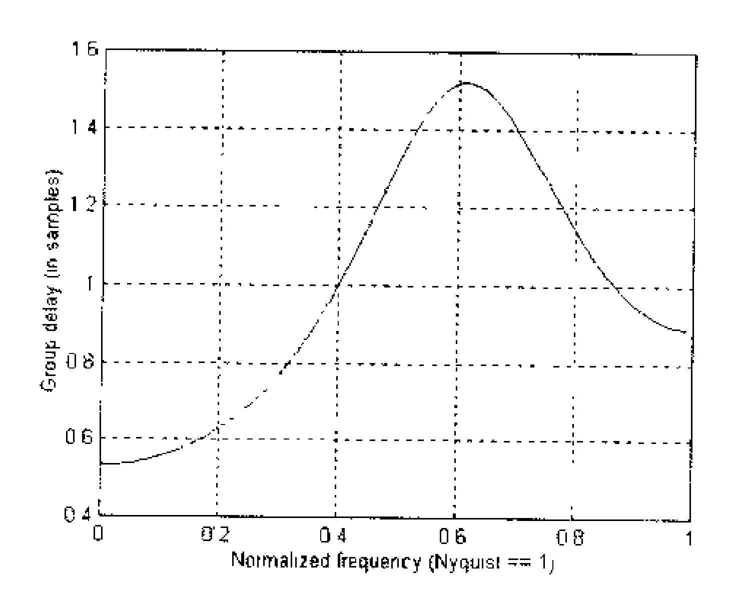


图 2.5 滤波器群延迟特性

12. impz

功能：数字滤波器的冲激响应。

格式：

```
[b, t]=impz(b, a)
[h, t]=impz(b, a, z)
[h, t]=impz(b, a, n, Fs)
impz(b, a)
```

说明：

由矢量 a 和 b 构成数字滤波器，即

$$H(z) = \frac{B(z)}{A(z)}$$

[h，t]=impz(b，a)计算出滤波器的冲激响应 h，取样点数 n 由 impz 函数自动选取，并记录在矢量 t 中(t=[0:n−1]')。

[h,t]=impz(b,a,n)可由用户指定取样点或取样时刻。当 n 为标量时,t=[0:n−1]',即在 0～n−1 时刻计算冲激响应,0 时刻表示滤波器的起始点;当 n 为矢量(其值应为整数),则表示 t=n,即在这些指定的时刻计算冲激响应。

[h, t]=impz(b, a, n, Fs)表示取样间隔为 1/Fs,在缺省 Fs 时,则取为 1。

不带输出变量的 impz 将在当前图形窗口中利用 stem(t, h)函数绘出冲激响应。

例 程序

```
b=[0.2  0.1  0.3  0.1  0.2];
a=[1  -1.1  1.5  -0.7  0.3];
impz(b, a, 50)
```

可得到如图 2.6 所示的滤波器的冲激响应。

参见:stem

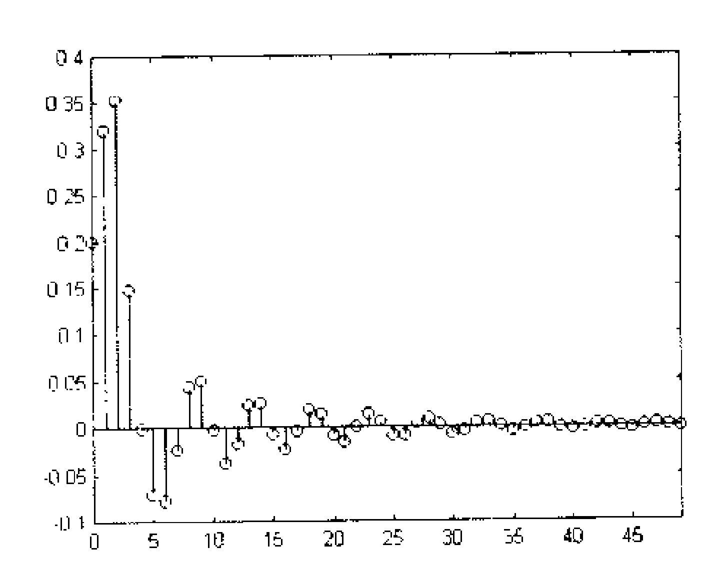


图 2.6 滤波器的冲激响应

13. zplane

功能:离散系统零极点图。

格式:

```
zplane(z, p)
zplane(b, a)
[hz, hp, ht]=zplane(z, p)
```

说明:

zplane 函数用于显示离散系统的零极点图。

zplane(z, p)可绘出列矢量 z 中的零点(以符号"o"表示)和列矢量 p 中的极点(以符号"×"表示),为提供参考还绘出了单位圆。如果 z 和 p 均为矩阵,则 zplane 以不同的颜色分别绘出 z 和 p 列中的零点和极点。

zplane 函数自动设定坐标刻度,以便绘出所有的零极点,但有时因有一个零极点数值

较大，可能使集中在原点邻域的零极点无法区分，为此可在 zplane 函数之后使用函数

```
axis([xmin  xmax  ymin  ymax])
```

或

```
set(gca, 'ylim', [ymin  ymax])
set(gca, 'xlim', [xmin  xmax])
```

来改写坐标刻度。

zplane(b, a)函数中，a，b 为行矢量，因此 zplane 函数首先利用 roots 函数找出由分子系数 b 和分母系数 a 构成的传递函数的零极点，然后再绘出零极点。

[hz, bp, ht]=zplane(z, p)可得到 3 个句柄矢量，其中 hz 为零点线句柄，hp 为极点线句柄，ht 为坐标轴、单位圆及文本对象的句柄。

例 输入程序

```
b=[0.2  0.1  0.3  0.1  0.2];
a=[1.0  -1.1  1.5  -0.7  0.3];
zplane(b, a)
```

可得到如图 2.7 所示的零极点图。

参见：freqz

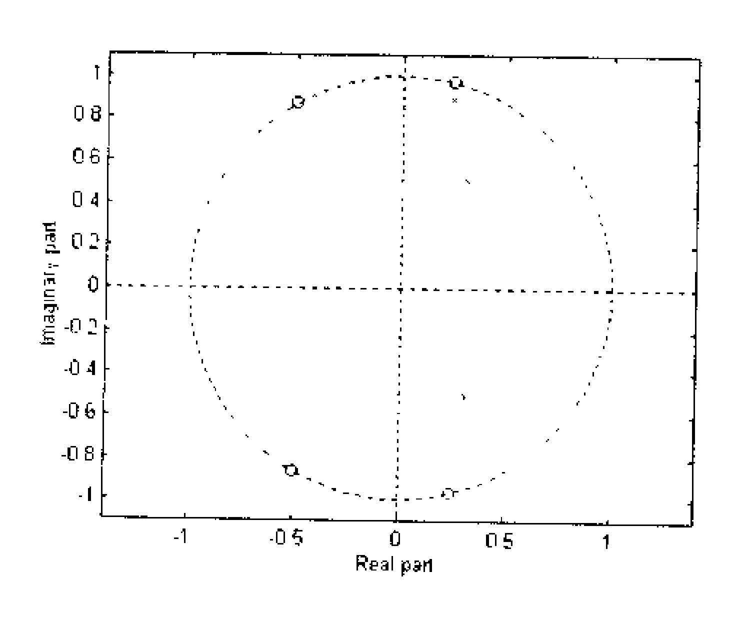


图 2.7 系统的零极点

2.3 线性系统变换

1. convmtx

功能：卷积矩阵。

格式：

```
A=convmtx(c, n)
```

A=convmtx(r，n)

说明：

卷积矩阵A是从某一矢量v形成的矩阵，其与任一矢量x的内积等于这两个矢量的卷积，即

(A，x)=conv(v，x)

在A=convmtx(c，n)中，c是长度为m的列矢量，则可产生(m+n−1)×n的矩阵A，且满足

(A，x)=conv(c，x)

其中x为任一列矢量。

同样，A=convmtx(r，n)中，r是长度为m的行矢量，则得到n×(m+n−1)的矩阵A，且满足

(A，x)=conv(r，x)

其中x为任一行矢量。

例 下列程序可产生一卷积矩阵

```
h=[1  2  3  2  1];
convmtx(h，7)
ans=
     1  2  3  2  1  0  0  0  0  0  0
     0  1  2  3  2  1  0  0  0  0  0
     0  0  1  2  3  2  1  0  0  0  0
     0  0  0  1  2  3  2  1  0  0  0
     0  0  0  0  1  2  3  2  1  0  0
     0  0  0  0  0  1  2  3  2  1  0
     0  0  0  0  0  0  1  2  3  2  1
```

参见： conv，deconv，dftmtx

2. poly2rc

功能： 从多项式系数中计算反射系数。

格式：

k=poly2rc(a)

说明：

k=poly2rc(a)可找出离散滤波器a网格结构的反射系数，a必须为实数，且a(1)≠0。反射系数可由下列递归过程求出

$$k(n)=a_n(n)$$

$$a_{n-1}(m)=\frac{a_n(m)-k(n)a_n(n-m)}{1-k(n)^2} \qquad m=1,2,\cdots,n-1$$

k是长度为length(a)−1的行矢量。

poly2rc函数的用途之一是用来判断多项式A(s)或A(z)的根是否位于单位圆内，这只需检查k的幅值是否小于1，从而判断其稳定性。

例 给定一 IIR 滤波器：

```
a=[1.000  0.6149  0.9899  0.0000  0.0031  -0.0082];
k=poly2rc(a)
k=
    0.3090  0.9800  0.0031  0.0082  -0.0082
```

应该注意一点，如果对任意的 i 都有 abs(k(i))=1，则计算反射系数为一病态条件问题，因此 poly2rc 函数会产生一些不确定值(NaN)，并给出警告信息。

参见：rc2poly

3. rc2poly

功能：从反射系数中计算多项式系数。

格式：

```
a=rc2poly(k)
```

说明：

a=rc2poly(k)可从反射系数 k 中找出首项为 1 的多项式系数，a 是长度为 length(k)+1 的行矢量。

例

```
k=[0.3090  0.9800  0.0031  0.0082  -0.0082];
a=rc2poly(k)
a=
   1.0000  0.6149  0.9899  0.0000  0.0031  -0.0082
```

参见：poly2rc

4. residuez

功能：Z 变换部分分式展开或留数计算。

格式：

```
[r, p, k]=residuez(b, a)
[b, a]=residuez(r, p, k)
```

说明：

residuez 函数可将以多项式之比表示的离散时间系统转化成部分分式展开或留数形式的系统，而且可进行相反转换。

[r, p, k]=residuez(b, a)可将以多项式之比 b(z)/a(z)转化成留数(r)、极点(p)和直接项(k)的部分分式展开。设

$$b(z) = b_0 + b_1 z^{-1} + \cdots + b_m z^{-m}$$

$$a(z) = a_0 + a_1 z^{-1} + \cdots + a_n z^{-n}$$

如果不存在重根，且 m>n-1，则有

$$\frac{b(z)}{a(z)} = \frac{r(1)}{1 - p(1)z^{-1}} + \cdots + \frac{r(n)}{1 - p(n)z^{-1}} + k(1) + \cdots + k(m - n + 1)z^{-(m+n)}$$

因此函数[r，p，k]=residuez(b，a)得到列矢量 r 为留数，列矢量 p 为极点位置，行矢量为直接项。极点数为

n=length(a)−1=length(r)=length(p)

如果 length(b)<length(a)，则直接项系数矢量 k 为空，否则

length(k)=length(b)−length(a)+1

如果 p(j)=p(j+1)=…=p(j+s−1)为 s 重极点，则分式中包含

$$\frac{r(j)}{1-p(j)z^{-1}}+\frac{r(j+1)}{(1-p(j)z^{-1})^2}+\cdots+\frac{r(j+s-1)}{(1-p(j)z^{-1})^s}$$

[b，a]=residuez(r，p，k)可将部分分式转化成多项式系数 a 和 b。

注意，residuez 函数与 MATLAB 环境中的 residue 函数非常类似，只不过，residue 函数适用于连续系统，而 residuez 函数适用于离散系统。

参见：ss2zp，tf2ss，zp2ss

5. sos2ss

功能：变系统二阶分割形式为状态空间形式。

格式：

[A，B，C，D]=sos2ss(sos)

说明：

离散传递函数的二阶分割形式为

$$H(z)=\prod_{k=1}^{L}H_k(z)=\prod_{k=1}^{L}\frac{b_{0k}+b_{1k}z^{-1}+b_{2k}z^{-2}}{a_{0k}+a_{1k}z^{-1}+a_{2k}z^{-2}}$$

其中系数由 sos 给出，sos 为 L×6 矩阵

$$sos=\begin{bmatrix} b_{01} & b_{11} & b_{21} & a_{01} & a_{11} & a_{21} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ b_{0L} & b_{1L} & b_{2L} & a_{0L} & a_{1L} & a_{2L} \end{bmatrix}$$

sos 矩阵必须为实矩阵。同样这一系统可表示成单输入单输出的状态方程

$$x(n+1)=Ax(n)+Bu(n)$$
$$y(n)=Cx(n)+Du(n)$$

其中 A 为 N×N 矩阵，N=2L−1，B 为 N−1 列矢量，C 为 N−1 行矢量，D 为标量。

根据这两种表示，可由 sos2ss 进行转换。

例 输入程序

```
sos=[1 2 3 4 0 -1; -2 1 -1 1 10 2];
[A, B, C, D]=sos2ss(sos)
```

可得

```
A=
    -10.2958    -2.2131         0         0
      2.2131          0         0         0
     -8.2958    -0.8576    0.2958    0.3195
           0          0    0.3195         0
```

B=

1

0

1

0

C=

4.1479 0.4288 0.1021 −0.9422

D=

−0.5000

参见：sos2tf，sos2zp，ss2sos，zp2sos

6. sos2tf

功能：变系统二阶分割形式为传递函数形式。

格式：

[num，den]=sos2tf(sos)

说明：

传递函数的二阶分割形式可参见 sos2ss 函数说明，传递函数可表示成

$$H(z)=\frac{num(z)}{den(z)}=\frac{num(1)+num(2)z^{-1}+\cdots+num(n+1)z^{-n}}{den(1)+den(2)z^{-1}+\cdots+den(m+1)z^{-m}}$$

[num，den]=sos2tf(sos)可将二阶分割形式变换成传递函数形式。

例

```
sos=[1 2 3 4 0 -1; -2 1 -1 1 10 2];
[num, den]=sos2tf(sos)
num=
      -2      -3      -5      1      -3
den=
       4      40       7    -10      -2
```

参见：sos2ss，sos2zp，ss2sos，zp2sos

7. sos2zp

功能：变系统二阶分割形式为零极点增益形式。

格式：

[z，p，k]=sos2zp(sos)

说明：

二阶分割形式可参见 sos2ss 函数的说明，系统的零极点增益表示为

$$H(z)=k\frac{(z-z(1))(z-z(2))\cdots(z-z(N))}{(z-p(1))(z-p(2))\cdots(z-p(M))}$$

[z，p，k]=sos2zp(sos)可将二阶分割形式变换成零极点增益形式。

例

```
sos=[1 2 3 4 0 -1; -2 1 -1 1 10 2];
[z, p, k]=sos2zp(sos)
z=
   -1.0000+1.4142i
   -1.0000-1.4142i
    0.2500+0.6614i
    0.2500-0.6614i
p=
    0.5000
   -0.5000
   -9.7958
   -0.2042
k=
   -0.5000
```

参见：sos2ss，sos2tf，ss2sos，zp2sos

8. ss2sos

功能：变系统状态空间形式为二阶分割形式。

格式：

```
sos=ss2sos(A, B, C, D)
sos=ss2sos(A, B, C, D, iu)
sos=ss2sos(A, B, C, D, 'order')
sos=ss2sos(A, B, C, D, iu, 'order')
```

说明：

有关系统的二阶分割形式和状态空间形式可参见 sos2ss 函数说明。

以(A，B，C，D)表示的系统必须是单输入实系统。如果系统是单输出系统，则 sos=ss2sos(A，B，C，D)可完成将状态空间形式变为二阶分割形式，它正好是[A，B，C，D]=sos2ss(sos)的逆过程。如果系统是多输出，则可采用 sos=ss2sos(A，B，C，D，iu)，其中标量 iu 用于指定变换中所使用的输出量。

sos=ss2sos(A，B，C，D，'order')和 sos=ss2sos(A，B，C，D，iu，'order')中，order 用于控制 sos 中的行顺序：

- 当 order=down 时，表示 sos 中第一行所包含的极点离单位圆最近；
- 当 order=up 时，表示 sos 中第一行所包含的极点离单位圆最远。

例 我们借助于 butter 函数产生系统的状态方程，然后由 ss2sos 函数求出二阶分割形式

```
[A, B, C, D]=butter(5, 0.2);
sos=ss2sos(A, B, C, D)
```

sos=

0.2453	0.2452	0	1.0000	−0.5095	0
0.0647	0.1294	0.0647	1.0000	−1.0966	0.3554
0.0808	0.1616	0.0808	1.0000	−1.3693	0.6926

参见：sos2ss，sos2tf，sos2zp，zp2sos

9. ss2tf

功能：变系统状态空间形式为传递函数形式。

格式：

[num，den]=ss2tf(A，B，C，D，iu)

说明：

系统的状态方程可表示为

$$x = Ax + Bu$$
$$y = Cx + Du$$

相应的传递函数为

$$H(s) = \frac{num(s)}{den(s)} = C(sI - A)^{-1}B + D$$

[num，den]=ss2tf(A，B，C，D，iu)可将状态空间表示变换成相应的传递函数表示，iu用于指定变换所使用的输入量。

ss2tf函数还可以应用于离散时间系统，这时得到的是Z变换表示的形式。

参见：ss2zp，tf2ss，tf2zp，zp2ss，zp2tf

10. ss2zp

功能：变系统状态空间形式为零极点增益形式。

格式：

[z，p，k]=ss2zp(A，B，C，D，iu)

说明：

系统状态空间表示和零极点增益表示可分别参考sos2ss和sos2zp函数说明。[z,p,k]=ss2zp(A，B，C，D，iu)可将状态空间表示转换成零极点增益表示，iu用于指示变换所用的输入量。

ss2zp函数还可以应用于离散时间系统，这时得到的是Z变换表示。

参见：ss2tf，tf2ss，zp2ss

11. tf2ss

功能：变系统传递函数形式为状态空间形式。

格式：

[A，B，C，D]=tf2ss(num，den)

说明：

tf2ss 函数可将给定系统的传递函数表示变换成等效的状态空间表示。在[A,B,C,D]=tf2ss(num，den)格式中，矢量 den 按 s 的降幂顺序输入分母系数，矩阵 num 每一行为相应于某输出的分子系数，其行数为输出的个数。tf2ss 得到控制器正则形式的 A、B、C、D 矩阵。

tf2ss 也可以用于离散系统中，但这时必须在分子多项式中补零，以使分子分母的长度相同。

例 将系统

$$H(s)=\frac{\begin{bmatrix}2s+3\\ s^2+2s+1\end{bmatrix}}{s^2+0.4s+1}$$

变换成状态空间表示

```
num=[0 2 3; 1 2 1];
den=[1 0.4 1];
[A, B, C, D]=tf2ss(num, den)
A=
    -0.4000     -1.0000
     1.0000           0
B=
     1
     0
C=
   2.0000     3.0000
   1.6000          0
D=
     0
     1
```

参见：ss2tf，ss2zp，zp2ss

12. tf2zp

功能：变系统传递函数形式为零极点增益形式。

格式：

```
[z, p, k]=tf2zp(num, den)
```

说明：

tf2zp 函数可找出多项式传递函数形式的系统的零点、极点和增益。

tf2zp 函数类似于 sos2zp、ss2zp 函数。

例 要找出系统

$$H(s)=\frac{s^2-0.5s+2}{s^2+0.4s+1}$$

的零点、极点和增益，可以输入

```
num=[1 -0.5 2];
den=[1 0.4 1];
[z, p, k]=tf2zp(num, den)
z=
   0.2500+1.3919i
   0.2500-1.3919i
p=
   -0.2000+0.9798i
   -0.2000-0.9798i
k=
    1
zplane(z, p)
```

同时利用 zplane 函数，得到如图 2.8 所示的零极点图。

参见：ss2tf，ss2zp，tf2ss，zp2tf

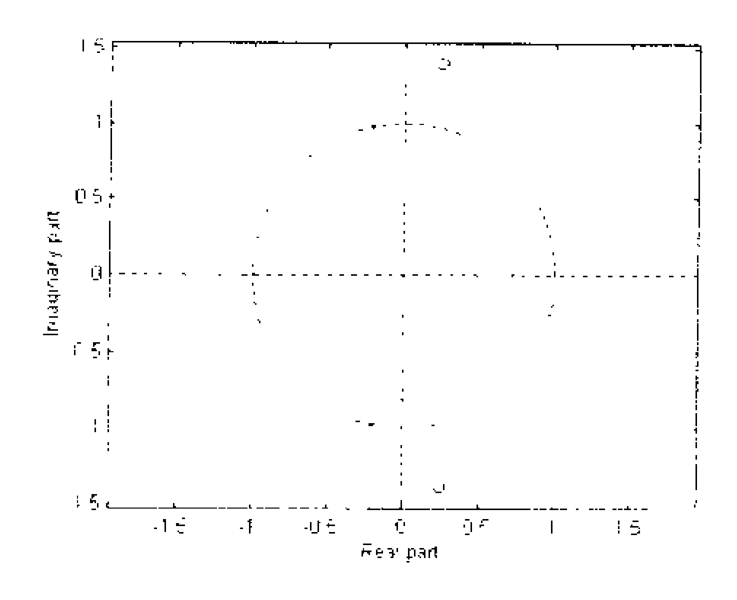


图 2.8　系统的零极点图

13. zp2sos

功能：变系统的零极点增益形式为二阶分割形式。

格式：

```
sos=zp2sos(z, p, k)
sos=zp2sos(z, p, k, 'order')
```

说明：

有关零极点增益形式和二阶分割形式分别参见 sos2zp 和 sos2ss 函数的说明。sos=zp2sos(z, p, k)正好完成[z, p, k]=sos2zp(sos)的逆过程。sos=zp2sos(z, p, k, 'order')

中，order 用于指定 sos 中的行顺序：

- order＝down 时，表示 sos 中第一行所包含的极点离单位圆最近；
- order＝up 时，表示 sos 中第一行所包含的极点离单位圆最远。

参见：sos2ss，sos2tf，sos2zp，ss2sos

14．zp2ss

功能：变系统零极点增益形式为状态空间形式。

格式：

```
[A, B, C, D]=zp2ss(z, p, k)
```

说明：

有关系统零极点增益形式和状态空间形式可参见 sos2zp 和 sos2ss 函数的说明。[A，B，C，D]＝zp2ss(z，p，k)可将以 z，p，k 表示的零极点增益形式变换成状态空间形式。

参见：ss2tf，ss2zp，tf2ss

15．zp2tf

功能：变系统零极点增益形式为传递函数形式。

格式：

```
[num, den]=zp2tf(z, p, k)
```

说明：

有关系统零极点增益表示和传递函数表示可参见 sos2zp 和 sos2tf 函数的说明。

参见：ss2tf，ss2zp，tf2ss，tf2zp，zp2ss

2.4 IIR 滤波器设计

1．besself

功能：Bessel(贝塞尔)模拟滤波器设计。

格式：

```
[b, a]=besself(n, Wn)
[b, a]=besself(n, Wn, 'ftype')
[z, p, k]=besself(…)
[A, B, C, D]=besself(…)
```

说明：

besself 函数可设计低通、带通、高通和带阻的模拟 Bessel 滤波器，其特性几乎完全由通过整个通带的群延迟确定，从而保持通带内滤波信号的波形。数字 Bessel 滤波器并不能保留这种特性，因此 besself 函数不支持数字 Bessel 滤波器的设计。

[b，a]＝besself(n，Wn)可设计出截止频率为 Wn 的 n 阶低通模拟滤波器(Wn>0)，得

到的滤波器传递函数为

$$H(s)=\frac{B(s)}{A(s)}=\frac{b(1)s^{n}+b(2)s^{n-1}+\cdots+b(n+1)}{s^{n}+a(2)s^{n-1}+\cdots+a(n+1)}$$

当 Wn 为二元矢量 Wn=[W1 W2](W1<W2)时，besself(n, Wn)设计出一个 2n 阶的模拟带通滤波器，其通带为 W1<ω<W2。

[b, a]=besself(n, Wn, 'ftype')可设计高通或带阻滤波器：

• 当 ftyep=high 时，可设计出截止频率为 Wn 的高通模拟滤波器；

• 当 ftype=stop 时，可设计带阻滤波器，这时 Wn=[W1 W2]，且阻带为 W1<ω<W2。

利用输出变量个数的不同，可得到滤波器的另外两种表示：零极点增益和状态方程。

• [z, p, k]=besself(n, Wn)或[z, p, k]=besself(n, Wn, 'ftype')可得到滤波器的零极点增益表示；

• [A, B, C, D]=besself(n, Wn)或[A, B, C, D]=besself(n, Wn, 'ftype')可得到滤波器的状态空间表示。

例 设计一 5 阶低通滤波器，截止频率为 1000 弧度/秒。其程序如下

```
[b, a]=besself(5, 1000);
freqs(b, a)
```

利用 freqs(b, a)函数可绘出低通滤波器的幅频、相频特性，如图 2.9 所示。

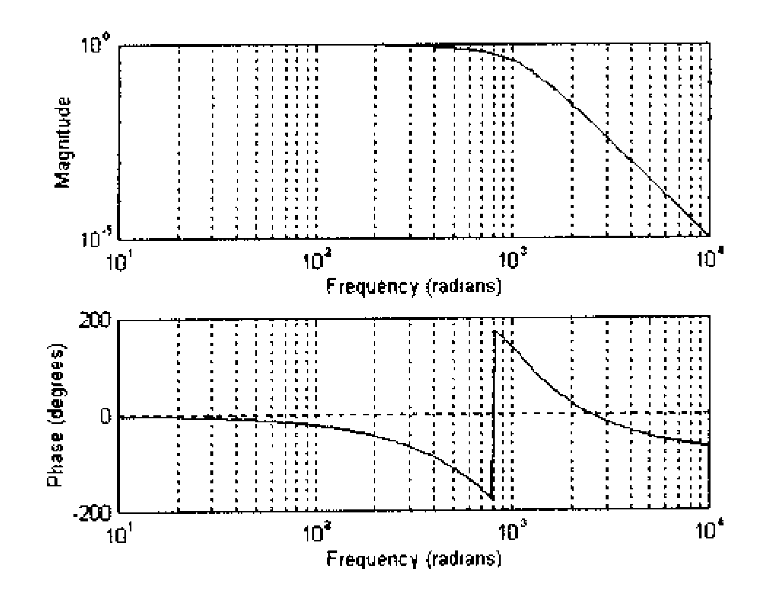


图 2.9 低通滤波器的幅频相频特性

例 要设计一 5 阶带通滤波器，通带为 2000<ω<10000，则可由程序

```
[b, a]=besself(6, [2000, 10000]);
Wn=logspace(0, 6, 300);
freqs(b, a, Wn)
```

得到带通滤波器特性如图 2.10 所示。

参见：besselap, butter, cheby1, cheby2, ellip

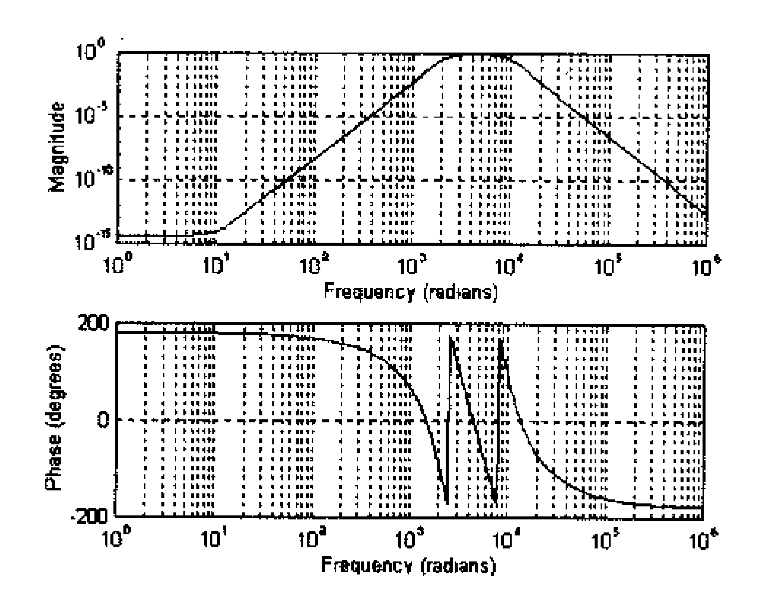


图 2.10 带通滤波器的幅频相频特性

2. butter

功能：Butterworth（比特沃思）模拟和数字滤波器设计。

格式：

```
[b, a]=butter(n, Wn)
[b, a]=butter(n, Wn, 'ftype')
[b, a]=butter(n, Wn, 's')
[b, a]=butter(n, Wn, 'ftype', 's')

[z, p, k]=butter(…)
[A, B, C, D]=butter(…)
```

说明：

butter 函数可设计低通、带通、高通和带阻的数字和模拟滤波器，其特性为使通带内的幅度响应最大限度地平坦，这会损失截止频率处的下降斜度。在期望通带平滑的情况下，可使用 butter 函数，但在期望下降斜度大的场合，应使用椭圆和 Chebyshev（切比雪夫）滤波器。

butter 函数可设计出数字域和模拟域的 Butterworth 滤波器。

（1）数字域

[b, a]=butter(n, Wn)可设计出截止频率为 Wn 的 n 阶低通 Butterworth 滤波器，其滤波器为

$$H(z)=\frac{B(z)}{A(z)}=\frac{b(1)+b(2)z^{-1}+\cdots+b(n+1)z^{-n}}{1+a(2)z^{-1}+\cdots+a(n+1)z^{-n}}$$

截止频率是滤波器幅度响应下降至 $1/\sqrt{2}$ 处的频率，但与 besself 函数不同，Wn∈[0, 1]，其中 1 相应于 $0.5f_s$（取样频率，即 Nyquist 频率）。

当 Wn=[W1 W2](W1<W2)时，butter 函数产生一 2n 阶的数字带通滤波器，其通带为 $W1<\omega<W2$。

[b，a]=butter(n，Wn，'ftype')可设计出高通或带阻滤波器：

• 当 ftype=high 时，可设计出截止频率为 Wn 的高通滤波器；

• 当 ftype=stop 时，可设计出带阻滤波器，这时 Wn=[W1 W2]，且阻带为 $W1<\omega<W2$。

利用输出变量个数的不同，可得到滤波器的另外两种表示：零极点增益和状态方程。

• [z，p，k]=butter(n，Wn)或[z，p，k]=butter(n，Wn，'ftype')可得到滤波器的零极点增益表示；

• [A，B，C，D]=butter(n，Wn)或[A，B，C，D]=butter(n，Wn，'ftype')可得到滤波器的状态空间表示。

(2) 模拟域

[b，a]=butter(n，Wn，'s')可设计截止频率为 Wn 的 n 阶低通模拟 Butterworth 滤波器为

$$H(s)=\frac{B(s)}{A(s)}=\frac{b(1)s^{n}+b(2)s^{n-1}+\cdots+b(n+1)}{s^{n}+a(2)s^{n-2}+\cdots+a(n+1)}$$

其中截止频率 Wn>0。

模拟域的 butter 函数说明与数字域的完全相同，它也有六种形式，限于篇幅，在此省略。

例 设数据采样频率为 900 Hz，现要设计一 9 阶的高通 Butterworth 滤波器，截止频率为 300 Hz。这是数字滤波器的设计问题，其程序为

```
[b，a]=butter(9，300/500，'high')；
freqz(b，a，128，1000)
```

结果如图 2.11 所示。

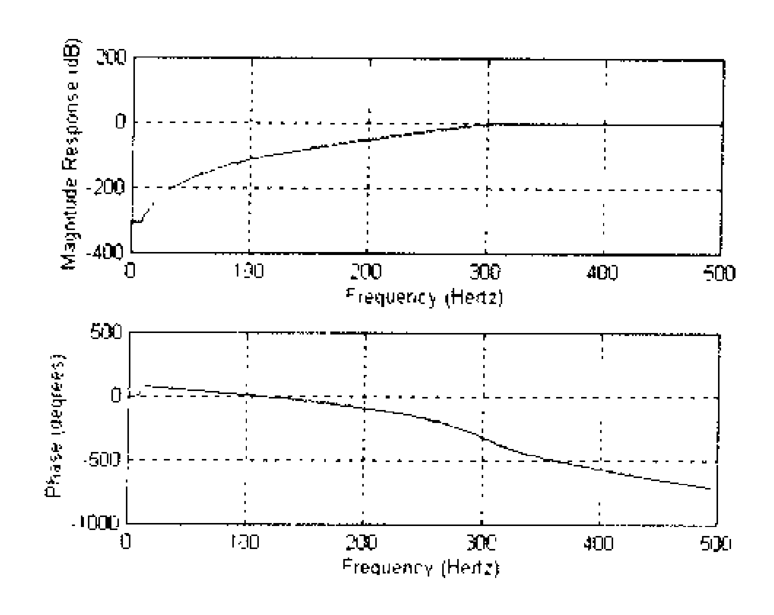


图 2.11 高通滤波器的幅频相频特性

例 设计一 10 阶的带通 Butterworth 滤波器，带通为 100～200 Hz，并画出滤波器的冲激响应。其程序如下

```
n=5;
Wn=[100 200]/500;
[b, a]=butter(n, Wn);
impz(b, a, 101)
```

结果如图 2.12 所示。

参见：besself，buttap，buttord，cheby1，cheby2，ellip

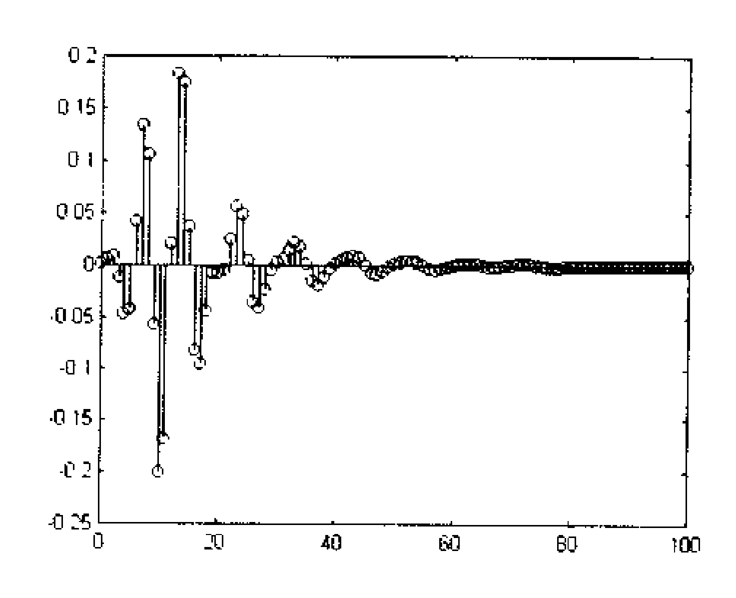


图 2.12 带通滤波器的冲激响应

3. cheby1

功能：Chebyshev(切比雪夫) I 型滤波器设计(通带等波纹)。

格式：

```
[b, a]=cheby1(n, Rp, Wn)
[b, a]=cheby1(n, Rp, Wn, 'ftype')
[b, a]=cheby1(n, Rp, Wn, 's')
[b, a]=cheby1(n, Rp, Wn, 'ftype', 's')

[z, p, k]=cheby1(…)
[A, B, C, D]=cheby1(…)
```

说明：

cheby1 函数可设计低通、带通、高通和带阻的数字和模拟 Chebyshev I 型滤波器，其通带内为等波纹，阻带内为单调。Chebyshev I 型滤波器的下降斜度比 II 型大，但其代价是在通带内波纹较大。

与 butter 函数类似，cheby1 函数可设计出数字域和模拟域的 Chebyshev Ⅰ型滤波器。

(1) 数字域

[b, a]=cheby1(n, Rp, Wn)可设计出 n 阶低通数字 Chebyshev Ⅰ型滤波器，其截止频率由 Wn 确定，通带内的波纹由 Rp(分贝)确定，滤波器为

$$H(z)=\frac{B(z)}{A(z)}=\frac{b(1)+b(2)z^{-1}+\cdots+b(n+1)z^{-n}}{1+a(2)z^{-1}+\cdots+a(n+1)z^{-n}}$$

截止频率是滤波器幅度下降至 $-$Rp 分贝处的频率，对 cheby1 函数，Wn∈[0, 1]，Wn=1 时相应于 0.5 f_s。通带波纹 Rp 越小，可得到更宽的变换宽度。

当 Wn=[W1 W2]时，cheby1 函数可产生一 2n 阶的数字带通滤波器，其通带为 W1<ω<W2。

[b, a]=cheby1(n, Rp, Wn, 'ftype')可设计高通或带阻滤波器：

· 当 ftype=high 时，可设计出截止频率为 Wn 的高通滤波器；

· 当 ftype=stop 时，可设计出带阻滤波器，这时 Wn=[W1 W2]，且阻带为 W1<ω<W2。

利用输出变量个数的不同，可得到滤波器的另外两种表示：零极点增益和状态方程。

· [z, p, k]=cheby1(n, Rp, Wn)或[z, p, k]=cheby1(n, Rp, Wn, 'ftype')可得到滤波器的零极点增益表示；

· [A, B, C, D]=cheby1(n, Rp, Wn)或[A, B, C, D]=cheby1(n, Rp, Wn, 'ftype')可得到滤波器的状态空间表示。

(2) 模拟域

[b, a]=cheby1(n, Rp, Wn, 's')可设计出截止频率为 Wn 的 n 阶低通模拟 Chebyshev Ⅰ型滤波器

$$H(s)=\frac{B(s)}{A(s)}=\frac{b(1)s^{n}+b(2)s^{n-1}+\cdots+b(n+1)}{s^{n}+a(2)s^{n-1}+\cdots+a(n+1)}$$

其中截止频率 Wn>0。

模拟域的 cheby1 函数说明与数字域完全相同，它也有六种形式，限于篇幅，在此省略。

例 与 butter 函数中的例 1 相似，现用 cheby1 函数来产生低通滤波器，并设定波纹系数为 0.5 dB。其程序如下

```
[b, a]=cheby1(9, 0.5, 300/500);
freqz(b, a, 512, 1000)
```

滤波器的幅频相频特性如图 2.13 所示。

例 与 butter 函数中的例 2 类似，现设计 10 阶 chebyshev Ⅰ带通滤波器，通带为 100 Hz<ω<200 Hz，要求画出滤波器的冲激响应。其程序如下

```
n=10; Rp=0.5;
Wn=[100 200]/500;
[b, a]=cheby1(n, Rp, Wn);
impz(b, a, 101)
```

得到如图 2.14 所示的滤波器冲激响应。

参见：besself，butter，cheblap，cheblord，cheby2，ellip

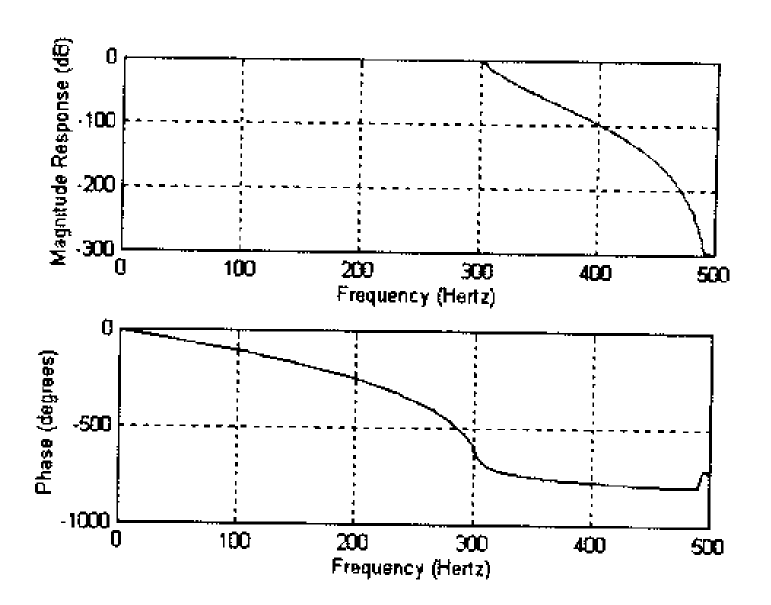


图 2.13　低通滤波器的幅频相频特性

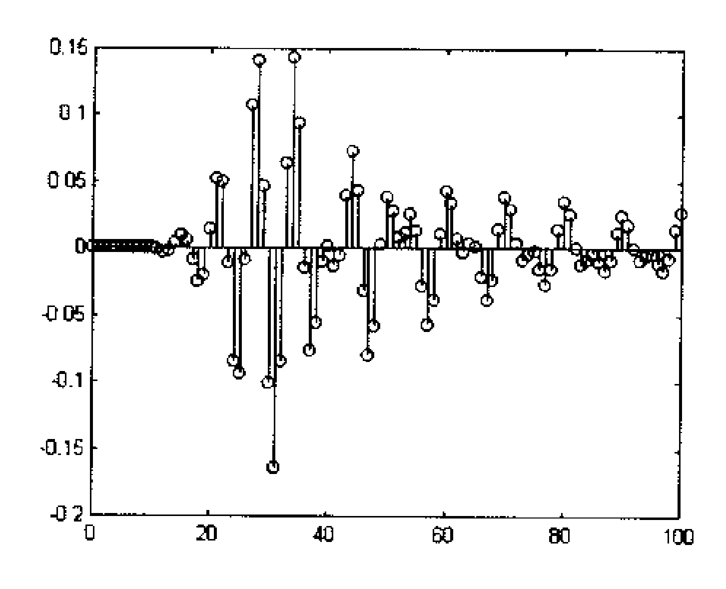


图 2.14　带通滤波器的冲激响应

4. cheby2

功能：Chebyshev Ⅱ型滤波器设计(阻带等波纹)。

格式：

[b，a]=cheby2(n，Rs，Wn)

[b，a]=cheby2(n，Rs，Wn，'ftype')

[b，a]=cheby2(n，Rs，Wn，'s')

```
[b, a]=cheby2(n, Rs, Wn, 'ftype', 's')

[z, p, k]=cheby2(…)
[A, B, C, D]=cheby2(…)
```

说明：

cheby2 函数与 cheby1 函数几乎是一模一样，只不过 cheby2 函数设计的滤波器其通带内为单调，阻带内为等波纹，因此由 Rs 指定阻带内的波纹。

cheby2 函数可设计低通、带通、高通和带阻的数字和模拟 chebyshev Ⅱ 型滤波器，针对所获得滤波器的形式：传递函数表示、零极点增益表示和状态方程，都有两种格式，因此对数字域和模拟域各有 6 种格式，详细说明可参见 cheby1 函数。这里不再赘述。

例 数据采样频率为 1000 Hz，设计一 9 阶 Chebyshev Ⅰ 型低通滤波器，其阻带比通带低 20 dB，截止频率为 300 Hz。其程序如下

```
[b, a]=cheby2(9, 20, 300/500);
freqz(b, a, 512, 1000)
```

滤波器的特性如图 2.15 所示。

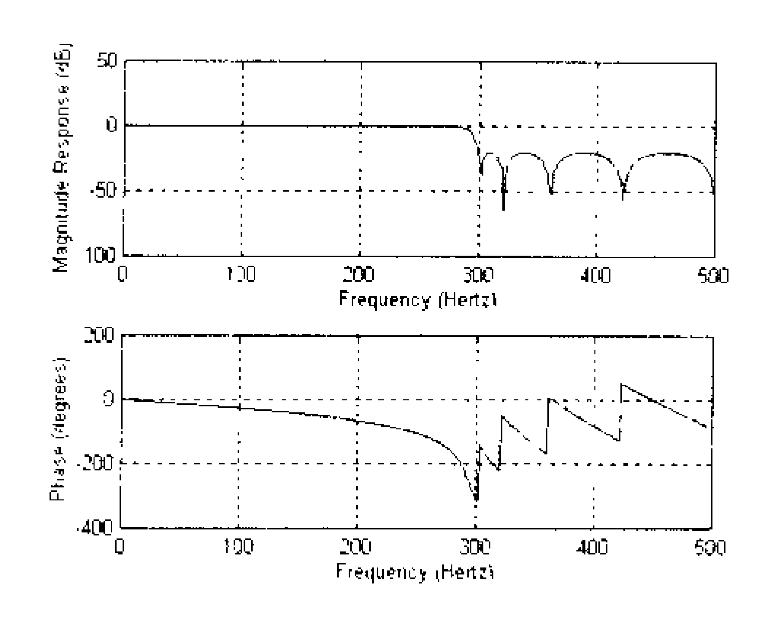


图 2.15 低通滤波器的幅频相频特性

例 2：设计 5 阶的带通 chebyshev Ⅰ 型滤波器，通带为 100 Hz<ω<200 Hz，阻带比通带低 20 dB，要求画出滤波器的冲激响应。其程序如下

```
n=5; Rs=20;
Wn=[100 200]/500;
[b, a]=cheby2(n, Rs, Wn);
impz(b, a, 101)
```

得到的滤波器冲激响应如图 2.16 所示。

参见：besself, butter, cheb2ap, cheb2ord, cheby1, ellip

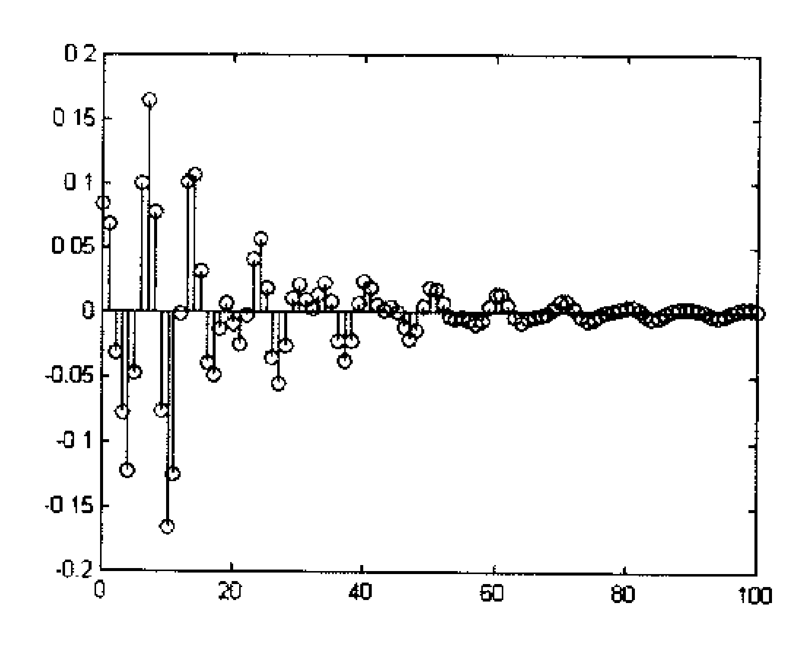


图 2.16　带通滤波器的冲激响应

5. ellip

功能：椭圆滤波器设计。

格式：

[b, a]=ellip(n, Rp, Rs, Wn)

[b, a]=ellip(n, Rp, Rs, Wn, 'ftype')

[b, a]=ellip(n, Rp, Rs, Wn, 's')

[b, a]=ellip(n, Rp, Rs, Wn, 'ftype', 's')

[z, p, k]=ellip(…)

[A, B, C, D]=ellip(…)

说明：

ellip 函数与 cheby1、cheby2 函数类似，它可设计低通、带通、高通和带阻的数字和模拟椭圆滤波器。与 Butterworth 或 Chebyshev 滤波器相比，ellip 函数可得到下降斜度更大的滤波器，但在通带和阻带内均为等波纹的。一般情况下，椭圆滤波器能以最低的阶实现指定的性能。

在[b, a]=ellip(n, Rp, Rs, Wn)中，Rp 用于指定通带的波纹，Rs 指定阻带波纹，Wn 指定截止频率。其它说明可参见 cheby1 函数，这里只作简要描述。

[b, a]=ellip(n, Rp, Rs, Wn)产生离散低通或带通(当 Wn=[W1 W2]时)滤波器。

[b, a]=ellip(n, Rp, Rs, Wn, 'ftype')产生离散高通(当 ftype=high 时)或带阻(当 ftype=stop，且 Wn=[W1 W2]时)滤波器。

[z, p, k]=ellip(n, Rp, Rs, Wn)或[z, p, k]=ellip(n, Rp, Rs, Wn, 'ftype')可产生以零极点增益形式表示的离散滤波器。

[A, B, C, D]=ellip(n, Rp, Rs, Wn)或[A, B, C, D]=ellip(n, Rp, Rs, Wn, 'ftype')可产生以状态空间表示的离散滤波器。

[b, a]=ellip(n, Rp, Rs, Wn, 's')或[b, a]=ellip(n, Rp, Rs, Wn, 'ftype', 's')可产生以传递函数表示的连续滤波器。

[z, p, k]=ellip(n, Rp, Rs, Wn, 's')或[z, p, k]=ellip(n, Rp, Rs, Wn, 'ftype', 's')可产生以零极点增益表示的连续滤波器。

[A, B, C, D]=ellip(n, Rp, Rs, Wn, 's')或[A, B, C, D]=ellip(n, Rp, Rs, Wn, 'ftype', 's')可产生以状态空间表示的连续滤波器。

例 设数据采样频率为 1000 Hz，现欲设计一 6 阶低通椭圆滤波器，其截止频率为 300 Hz，通带波纹为 3 dB，阻带波纹为 50 dB。其程序如下

```
[b, a]=ellip(6, 3, 50, 300/500);
freqz(b, a, 512, 1000)
```

得到如图 2.17 所示的滤波器幅频相频特性。

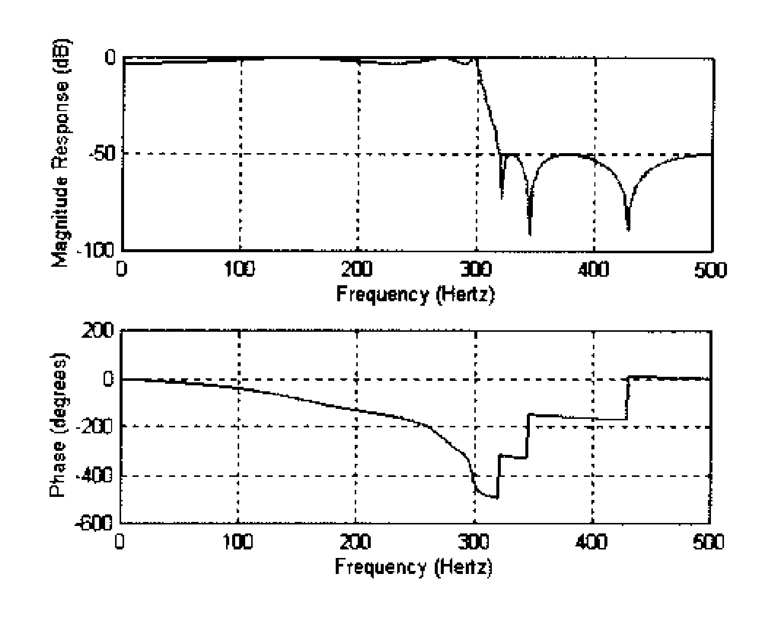


图 2.17 低通滤波器特性

例 设计一 20 阶的带通椭圆滤波器，其通带为 100 Hz<ω<200 Hz，并画出滤波器的冲激响应。其程序如下

```
[b, a]=ellip(10, 0.5, 20, [100 200]/500);
impz(b, a, 101)
```

这可得到如图 2.18 所示的滤波器冲激响应。

参见：besself, butter, cheby1, cheby2, ellipord, ellipap

6. yulewalk

功能：递归数字滤波器设计。

格式：

[b, a]=yulewalk(n, f, m)

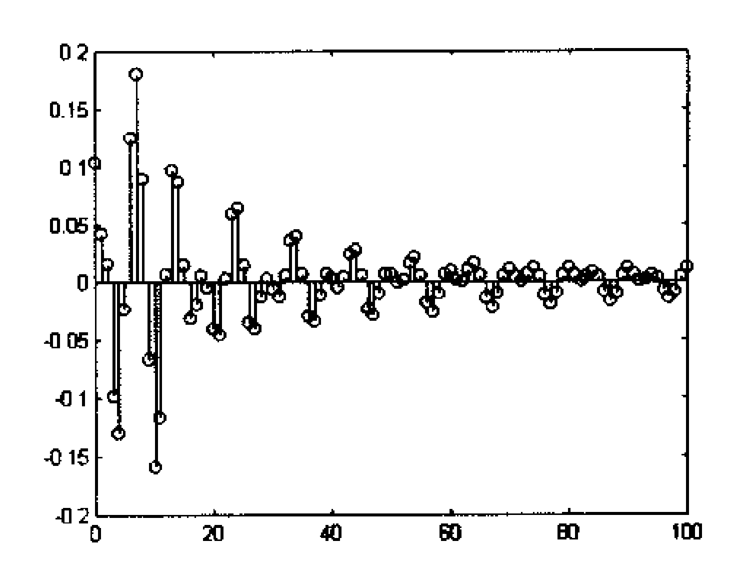


图 2.18 带通滤波器的冲激响应

说明：

yulewalk 函数利用对指定的频率响应进行最小二乘拟合的方法来设计递归的 IIR 数字滤波器。

[b, a]=yulewalk(n, f, m)可得到以 b 和 a 表示的 n 阶 IIR 滤波器，其幅频特性与由矢量 f 和 m(幅值)所给定的特性近似匹配：

- f 为频率点矢量，且 f∈[0, 1]，当 f=1 时就相应于 0.5 f_s。矢量 f 中按升序排列，且第一个必须为 0，最后一个必须为 1，并允许出现相同的频率值；
- 矢量 m 中包含与 f 相对应的期望滤波器响应幅度；
- 矢量 f 和 m 的长度必须相同；
- plot(f, m)可画出滤波器的幅频特性。

由[b, a]=yulewalk(n, f, m)得到的滤波器可写成

$$H(z)=\frac{B(z)}{A(z)}=\frac{b(1)+b(2)z^{-1}+\cdots+b(n+1)z^{-n}}{1+a(2)z^{-1}+\cdots+a(n+1)z^{-n}}$$

在指定滤波器频率特性时，应尽量避免从通带到阻带之间陡峭的过度过程，应选择一定的斜度。

例 设计一 8 阶的低通滤波器，并画出期望的频率响应和实际的频率响应。其程序如下

```
f=[0  0.6  0.65  0.7  1];
m=[1  1  0.5  0  0];
[b, a]=yulewalk(8, f, m);
[h, w]=freqz(b, a, 128);
plot(f, m, w/pi, abs(h), '--')
```

结果如图 2.19 所示。

参见： butter, cheby1, cheby2, ellip, fir2, remz

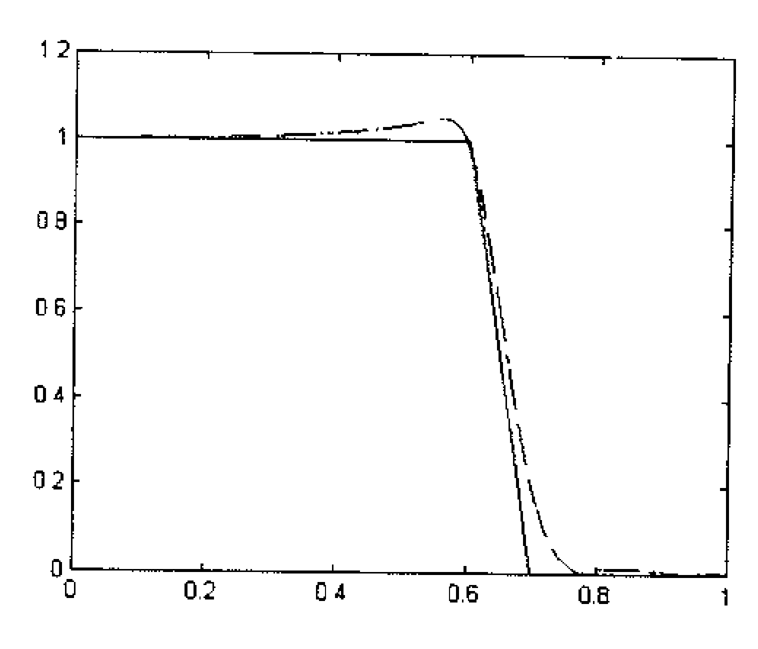


图 2.19 理想和实际滤波器幅频特性

2.5 IIR 滤波器阶的选择

1. buttord

功能：Butterworth 滤波器阶的选择。

格式：

```
[n, Wn]=buttord(Wp, Ws, Rp, Rs)
[n, Wn]=buttord(Wp, Ws, Rp, Rs, 's')
```

说明：

buttord 可在给定滤波器性能情况下，选择模拟或数字 Butterworth 滤波器最小的阶，其中，Wp 和 Ws 分别是通带和阻带的拐角频率(截止频率)，其值为 0≤Wp(或 Ws)≤1，当其值为 1 时表示 0.5 f_s。Rp 和 Rs 分别是通带和阻带区的波纹系数。

(1) 数字域

[n, Wn]=butter(Wp, Ws, Rp, Rs)可得到数字 Butterworth 滤波器的最小阶 n，并使在通带(0, Wp)内波纹系数小于 Rp，在阻带(Ws, 1)内衰减系数大于 Rs。buttord 还得到了截止频率 Wn，这样利用 butter 函数可产生满足指定性能的滤波器。

利用 buttord 函数，还可以得到高通、带通和带阻滤波器的阶。当 Wp>Ws 时，这时为高通滤波器；当 Wp，Ws 为二元矢量时，若 Wp<Ws，则为带通或带阻滤波器，这时 Wn 也为二元矢量。

(2) 模拟域

[n, Wn]=buttord(Wp, Ws, Rp, Rs, 's')可得满足指定性能的模拟 Butterworth 滤波器的阶 n 和截止频率 Wn，从而可利用 butter 函数设计模拟滤波器。

例 设计一低通滤波器，通带范围 0～100 Hz，通带波纹小于 3 dB，阻带为－30 dB，并利用最小的阶来实现。其程序如下

```
Wp=100/500; Ws=200/500;
[n, Wn]=buttord(Wp, Ws, 3, 30);
[b, a]=butter(n, Wn);
freqz(b, a, 512, 1000)
```

从而得到如图 2.20 所示的低通滤波器特性。

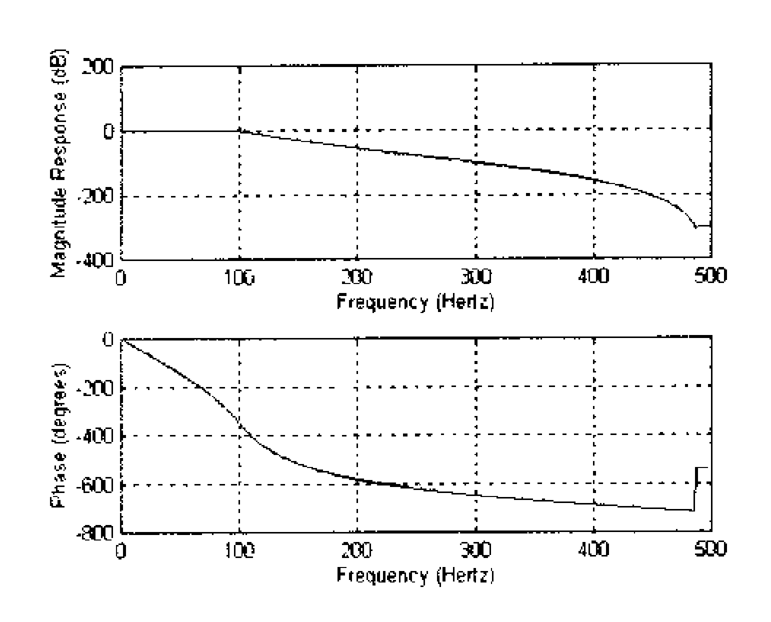


图 2.20 低通滤波器特性(n=8)

例 设计一带通滤波器,通带范围为 100～250 Hz,其余指标同例 1。其程序如下

```
Wp=[100 250]/500; Ws=[50 300]/500;
[n, Wn]=buttord(Wp, Ws, 3, 30);
[b, a]=butter(n, Wn);
freqz(b, a, 512, 1000)
```

得到如图 2.21 所示的带通滤波器特性。

例 设计与例 2 条件相同的带阻滤波器。其程序如下

```
Wp=[100 250]/500; Ws=[50 250]/500;
[n, Wn]=buttord(Wp, Ws, 3, 30);
[b, a]=butter(n, Wn);
freqz(b, a, 512, 1000)
```

得到如图 2.22 所示的带阻滤波器特性。

参见：butter, cheb1ord, cheb2ord, ellipord

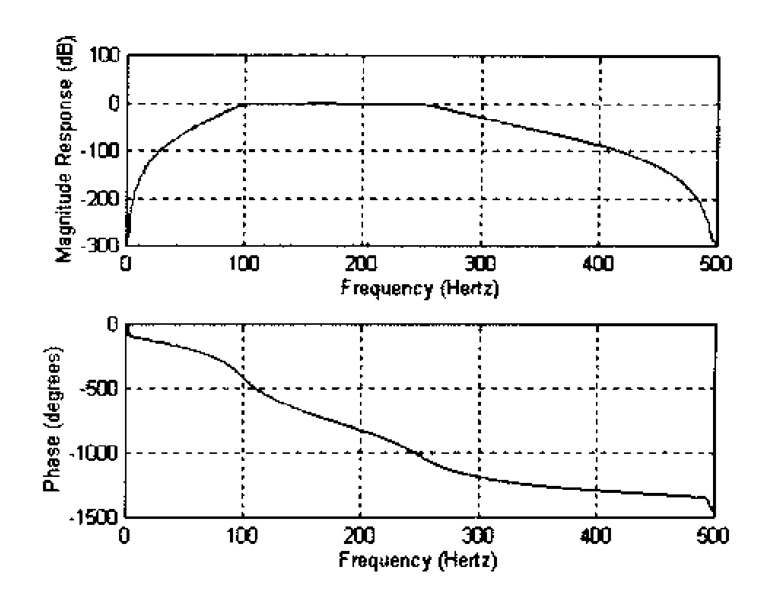


图 2.21 带通滤波器特性(n=7)

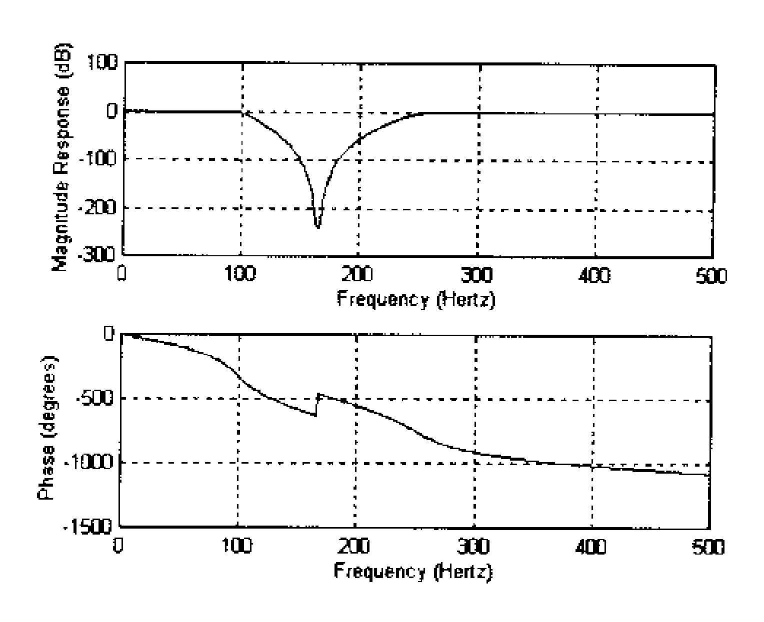


图 2.22 带阻滤波器特性(n=7)

2. cheb1ord

功能：Chebyshev Ⅰ 型滤波器阶的选择。

格式：

[n, Wn]=cheb1ord(Wp, Ws, Rp, Rs)

[n, Wn]=cheb1ord(Wp, Ws, Rp, Rs, 's')

说明：

在给定滤波器性能的情况下，选择 Chebyshev Ⅰ 型滤波器的最小阶，其中 Wp 和 Ws 分别是通带和阻带的拐角频率(截止频率)，其值为 0≤Wp(或 Ws)≤1，Rp 和 Rs 分别是通

带和阻带区的波纹系数。

(1) 数字域

[n, Wn]=cheb1ord(Wp, Ws, Rp, Rs)可得到数字 Chebyshev I 型滤波器的最小阶，并使其在通带(0, Wp)内波纹系数小于 Rp，在阻带(Ws, 1)内衰减系数大于 Rs，cheb1ord 还得到了截止频率 Wn，这样利用 cheby1 函数可得到满足指定性能的滤波器。

cheb1ord 函数还可以得到高通、带通和带阻滤波器的阶。

(2) 模拟域

[n, Wn]=cheb1ord(Wp, Ws, Rp, Rs, 's')可得到满足指定性能的模拟 Chebyshev I 型滤波器的阶 n 和截止频率 Wn，从而利用 cheby1 函数设计模拟滤波器。

例 设计一低通 Chebyshev I 型滤波器，通带范围 0~100 Hz，通带波纹 3 dB，阻带衰减－30 dB，数据采样频率为 1 000 Hz。

首先利用 cheb1ord 函数找出最小阶，然后利用 cheby1 函数来实现，其程序如下

```
Wp=100/500; Ws=200/500;
[n, Wn]=cheb1ord(Wp, Ws, 3, 30);
[b, a]=cheby1(n, 3, Wn);
freqz(b, a, 512, 1000)
```

这样就得到了如图 2.23 所示的低通滤波器特性。

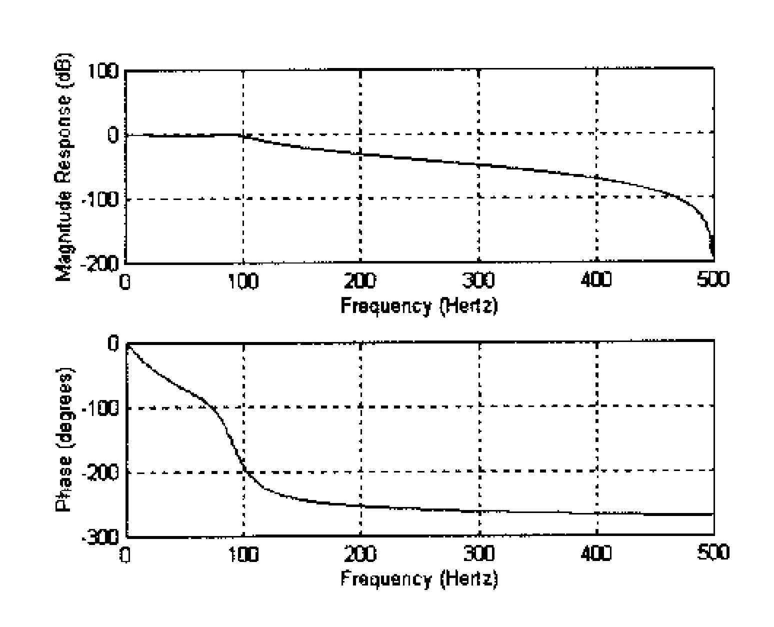


图 2.23 低通滤波器特性(n=3)

例 设计一带通滤波器，通带范围为 100~250 Hz，其余指标同例 1。其程序如下

```
Wp=[100 250]/500; Ws=[50 300]/500;
[n, Wn]=cheb1ord(Wp, Ws, 3, 30);
[b, a]=cheby1(n, 3, Wn);
freqz(b, a, 512, 1000)
```

这就得到了如图 2.24 所示的带通滤波器特性。

参见：cheby1, buttord, cheb2ord, ellipord

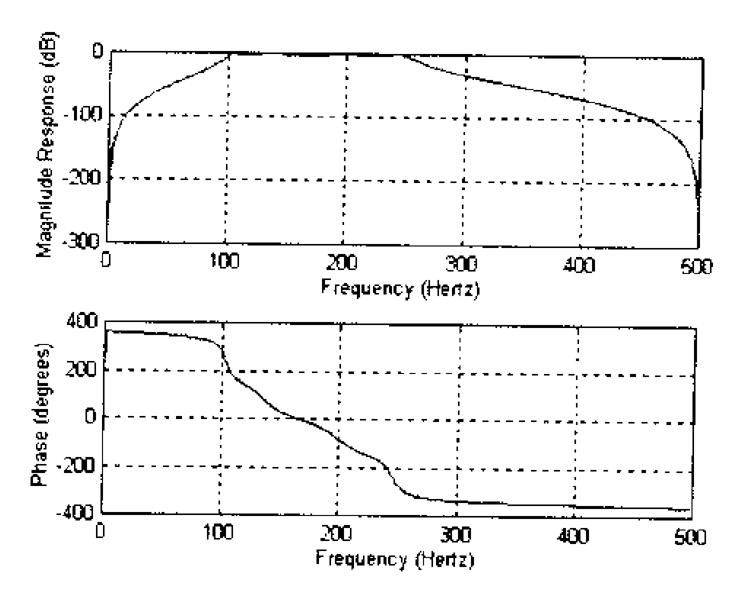


图 2.24 带通滤波器特性(n=4)

3. cheb2ord

功能：Chebyshev Ⅱ型滤波器阶的选择。

格式：

```
[n, Wn]=cheb2ord(Wp, Ws, Rp, Rs)
[n, Wn]=cheb2ord(Wp, Ws, Rp, Rs, 's')
```

说明：

cheb2ord 函数与 cheb1ord 函数类似，只不过它用于选择指定性能时的 Chebyshev Ⅱ型滤波器的阶 n 和截止频率 Wn，与 cheby2 函数配合可设计出最低阶的 Chebyshev Ⅱ型滤波器，详见 cheb1ord 函数。

例 要求与 cheb1ord 函数中的例 1 相同。其程序也类似如下

```
Wp=100/500; Ws=200/500;
[n, Wn]=cheb2ord(Wp, Ws, 3, 30);
[b, a]=cheby2(n, 3, Wn);
freqz(b, a, 512, 1000)
```

这样就得到了如图 2.25 所示的低通滤波器特性。读者可与图 2.23 作一比较，不难发现两者的差异。

例 要求与 cheb1ord 函数中的例 2 相同。其程序也类似如下

```
Wp=[100 250]/500; Ws=[50 300]/500;
[n, Wn]=cheb2ord(Wp, Ws, 3, 30);
[b, a]=cheby2(n, 3, Wn);
freqz(b, a, 512, 1000)
```

这就得到了如图 2.26 所示的带通滤波器特性。

参见：cheby2，cheb1ord，buttord，ellipord

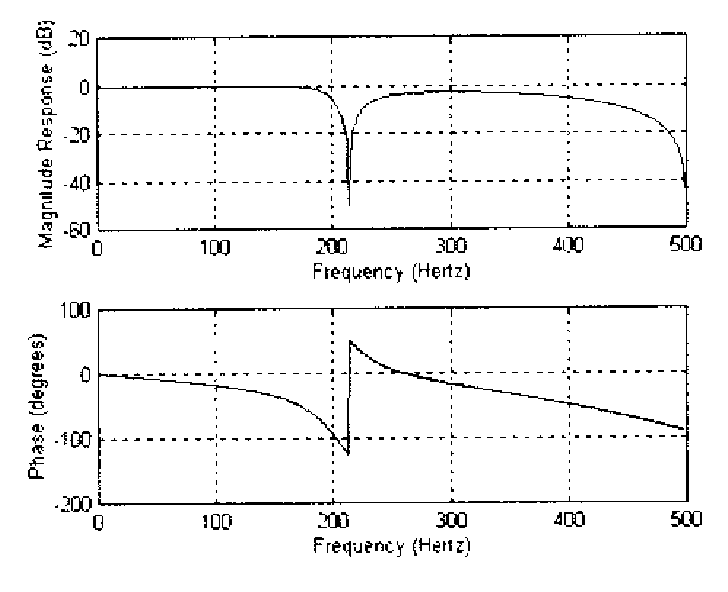


图 2.25 低通滤波器特性(n=3)

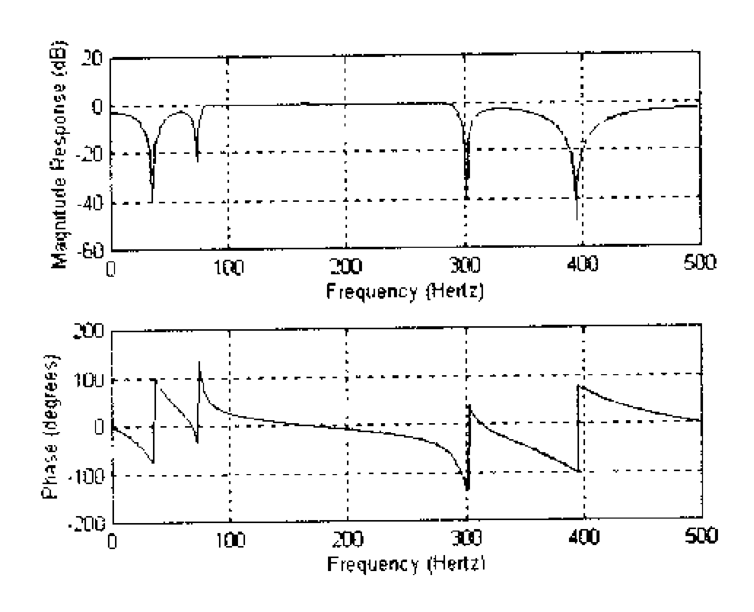


图 2.26 带通滤波器特性(n=4)

4. ellipord

功能：椭圆滤波器阶的选择。

格式：

[n, Wn]=ellipord(Wp, Ws, Rp, Rs)

[n, Wn]=ellipord(Wp, Ws, Rp, Rs, 's')

说明：

ellipord 函数与 cheb1ord 函数类似，只不过它用于选择指定性能时的椭圆滤波器的阶 n 和截止频率 Wn，并与 ellip 函数配合可设计出最低阶的椭圆滤波器。这里省略说明，详见 cheb1ord 函数。

例 要求与 cheb1ord 函数中的例 1 相同。其程序也类似如下

Wp=100/500; Ws=200/500;

```
[n, Wn]=ellipord(Wp, Ws, 3, 30);
[b, a]=ellip(n, 3, 30, Wn);
freqz(b, a, 512, 1000)
```

这样就得到了如图 2.27 所示的低通滤波器特性。读者可与图 2.23、图 2.25 作一比较，不难发现它们之间的差异。

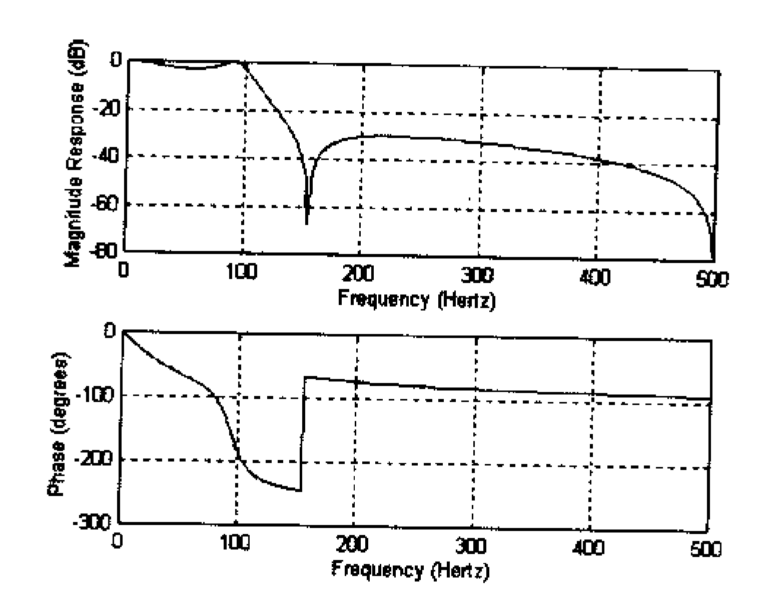


图 2.27　低通滤波器特性(n=3)

例　要求与 cheb1ord 函数中的例 2 相同。其程序也类似如下

```
Wp=[100 250]/500; Ws=[50 300]/500;
[n, Wn]=ellipord(Wp, Ws, 3, 30);
[b, a]=ellip(n, 3, 30, Wn);
freqz(b, a, 512, 1000)
```

这就得到了如图 2.28 所示的带通滤波器特性。

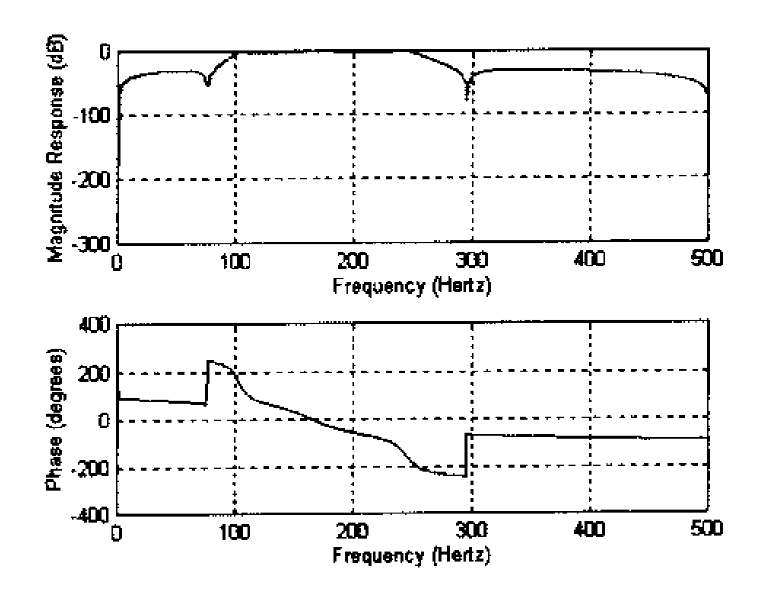


图 2.28　带通滤波器特性(n=3)

参见：ellip，buttord，cheb1ord，cheb2ord

2.6 FIR 滤波器设计

1. fir1

功能：基于窗函数的 FIR(有限冲激响应)滤波器设计——标准频率响应。

格式：

```
b=fir1(n, Wn)
b=fir1(n, Wn, 'ftype')
b=fir1(n, Wn, Window)
b=fir1(n, Wn, 'ftype', Window)
```

说明：

fir1 函数以经典方法实现加窗线性相位 FIR 数字滤波器设计，它可设计出标准的低通、带通、高通和带阻滤波器(具有任意频率响应的加窗滤波器由 fir2 函数设计)。

b=fir1(n，Wn)可得到 n 阶低通 FIR 滤波器，滤波器系数包含在 b 中，这可表示成

$$b(z) = b(1) + b(2)z^{-1} + \cdots + b(n+1)z^{-n}$$

这是一个截止频率为 Wn 的 Hamming(汉明)加窗线性相位滤波器，$0 \leqslant Wn \leqslant 1$，Wn=1 相应于 0.5 f_s。

当 Wn=[W1 W2]时，fir1 函数可得到带通滤波器，其通带为 $W1 < \omega < W2$。

b=fir1(n，Wn，'ftype')可设计高通和带阻滤波器，由 ftype 决定：

- 当 ftype=high 时，设计高通 FIR 滤波器；
- 当 ftype=stop 时，设计带阻 FIR 滤波器。

在设计高通和带阻滤波器时，fir1 函数总是使用阶为偶数的结构，因此当输入的阶次为奇数时，fir1 函数会自动将阶次加 1。这是因为对奇次阶的滤波器，其在 Nyquist 频率处的频率响应为零，因此不适合于构成高通和带阻滤波器。

b=fir1(n，Wn，Window)则利用列矢量 Window 中指定的窗函数进行滤波器设计，Window 长度为 n+1。如果不指定 Window 参数，则 fir1 函数采用 Hamming 窗。

b=fir1(n，Wn，'ftype'，Window)可利用 ftype 和 Window 参数，设计各种加窗的滤波器。

由 fir1 函数设计的 FIR 滤波器的群延迟为 n/2。

例 设计一 24 阶 FIR 带通滤波器，通带为 $0.35 < \omega < 0.65$。其程序如下

```
b=fir1(48, [0.35  0.65]);
freqz(b, 1, 512)
```

这可得到如图 2.29 所示的带通 FIR 滤波器特性。

例 设计一 34 阶的高通 FIR 滤波器，截止频率为 0.48，并使用具有 30 dB 波纹的 Chebyshev 窗。其程序如下

```
Window=chebwin(35, 30);
b=fir1(34, 0.48, 'high', Window);
freqz(b, 1, 512)
```

这时，得到如图 2.30 所示的高通 FIR 滤波器特性。

参见：fir2，fir1s，butter，cheby1，cheby2，remez，yulewalk，ellip

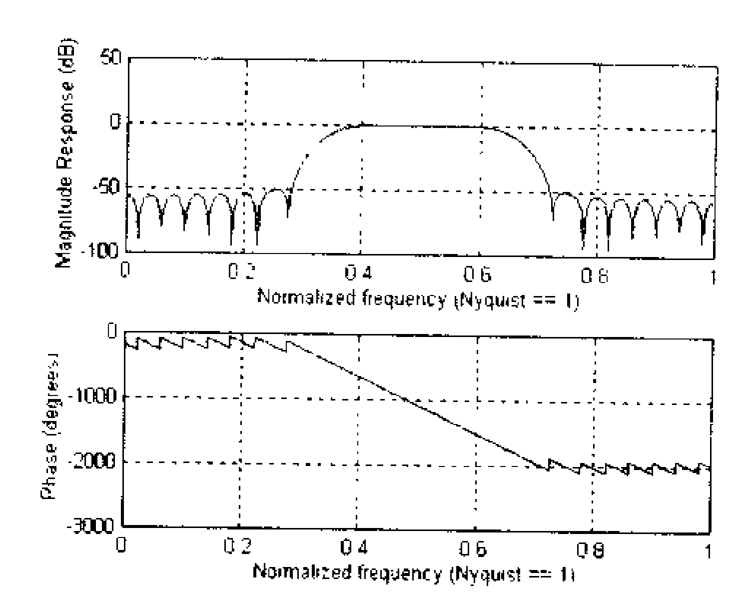


图 2.29　带通 FIR 滤波器特性

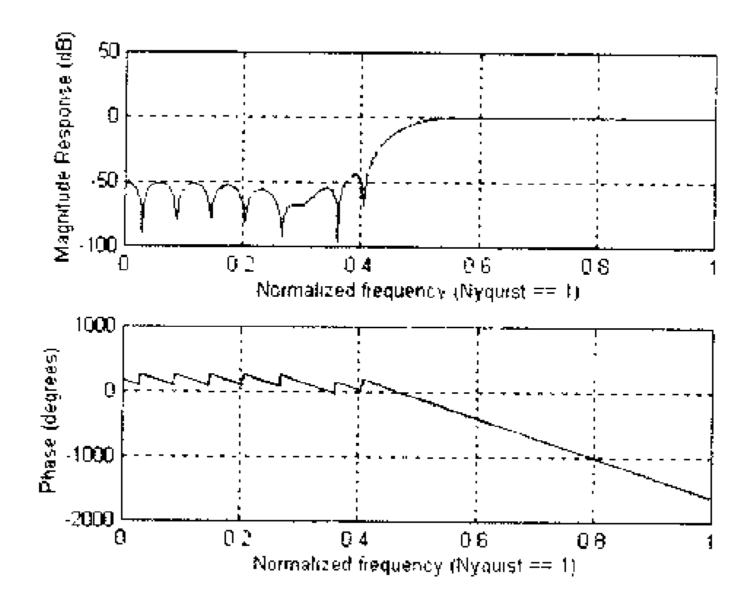


图 2.30　高通 FIR 滤波器特性

2. fir2

功能：基于窗函数的 FIR 滤波器设计——任意频率响应。

格式：

```
b=fir2(n, f, m)
b=fir2(n, f, m, Window)
```

b=fir2(n, f, m, npt)

b=fir2(n, f, m, npt, Window)

b=fir2(n, f, m, npt, lap)

b=fir2(n, f, m, npt, lap, Window)

说明:

fir2 函数用于设计具有任意频率响应的加窗数字 FIR 滤波器，对标准的低通、带通、高通和带阻滤波器的设计可采用 fir1 函数。

b=fir2(n, f, m)可设计出一 n 阶的 FIR 滤波器，其滤波器的频率特性由矢量 f 和 m 决定，有关 f 和 m 的约定可参见 yulewalk 函数。

b=fir2(n, f, m, Window)可将列矢量 Window 中指定的窗函数用于滤波器设计，如省略 Window，则自动选取 Hamming 窗。

b=fir2(n, f, m, npt)格式中，可利用参数 npt 指定 fir2 对频率响应进行内插的点数，对应的 b=fir2(n, f, m, npt, Window)格式中，可指定窗函数。

b=fir2(n, f, m, npt, lap)格式中，可利用参数 lap 指定 fir2 在重复频率点附近插入的区域大小，对应的 b=fir2(n, f, m, npt, lap, Window)格式中，可指定窗函数。

例 设计一 30 阶的低通 FIR 滤波器，使之与期望频率特性相近。其程序如下

```
f=[0  0.6  0.6  1];
m=[1  1  0  0];
b=fir2(30, f, m);
[h, w]=freqz(b, 1, 128);
plot(f, m, w/pi, abs(h))
```

结果如图 2.31 所示。

参见: fir1, butter, cheby1, cheby2, ellip, remez, yulewalk

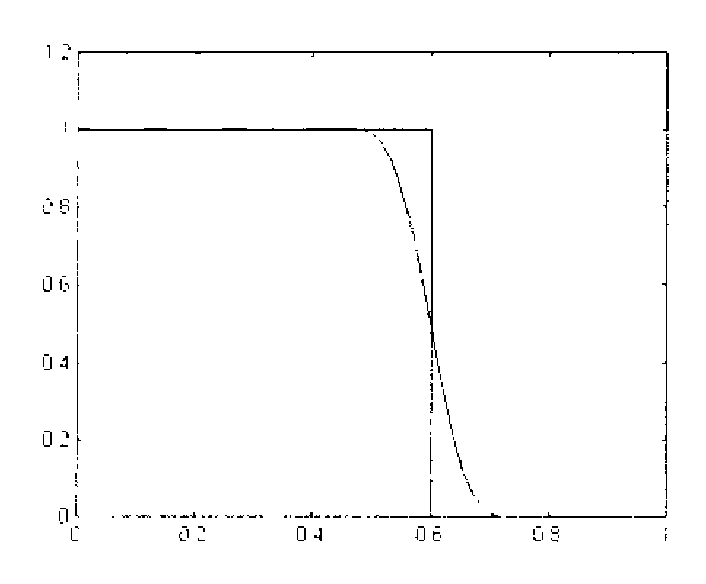


图 2.31 理想和实际滤波器特性

3. firls

功能：最小二乘线性相位 FIR 滤波器设计。

格式：

```
b=firls(n, f, m)
b=firls(n, f, m, w)
b=firls(n, f, m, 'ftype')
b=firls(n, f, m, w, 'ftype')
```

说明：

firls 函数设计出的线性相位 FIR 滤波器，可使指定频段内的理想分段线性函数与滤波器幅度响应之间的误差平方和最小。

b=firls(n，f，m)可设计出 n 阶 FIR 滤波器，其幅频特性匹配于由 f 和 m 给出的特性，矢量 b 中包含有 n+1 个系数，这些系数遵循对称关系为

$$b(k) = b(n + 2 - k) \quad k = 1, \cdots, n + 1$$

这些滤波器就是 I 型(n 为奇数)和 II 型(n 为偶数)线性相位滤波器。有关矢量 f 和 m 的说明可参见 yulewalk 函数，但这里有一点是不同的：期望的幅频值介于(f(k)，f(k+1))(k 为奇数)之间，即为连接点(f(k)，m(k))与(f(k+1)，m(k+1))的线段，而在(f(k+1)，f(k+2))区间为非指定区，即“无关”区。

b=firls(n，f，m，w)可利用权值矢量 w 对各频率段进行加权拟合，w 的长度为 f 和 m 长度的一半。

b=firls(n，f，m，'ftype')和 b=firls(n，f，m，w，'ftype')可指定滤波器的类型：

• 当 fytpe=hilbert 时，设计的滤波器为奇对称的线性相位滤波器(III 型和 IV 型)，b 的系数满足 b(k)=−b(n+2−k)，k=1，2，…，n+1。这类滤波器包括 Hilbert(希尔伯特)变换器。

• 当 ftype=differentiator 时，也设计出 III 型和 IV 型滤波器，但采用了特殊的加权技术，对非零幅值的频段，对误差平方和实施$(1/f)^2$的加权，从而使低频段误差大大小于高频段误差。

例 设计一 24 阶的反对称滤波器，使其具有分段线性通带。画出理想和实际的频率响应。其程序如下

```
f=[0  .3  .4  .6  .7  .9];
m=[0  1  0  0  0.5  0.5];
b=firls(24, f, m, 'hilbert')
for i=1:2:6
    plot([f(i) f(i+1)], [m(i) m(i+1)], '--'), hold on
end
[h, f]=freqz(b, 1, 512, 2);
plot(f, abs(h))
grid
```

这样，就得到了如图 2.32 所示的滤波器特性。

参见：fir1，fir2，remez，yulewalk

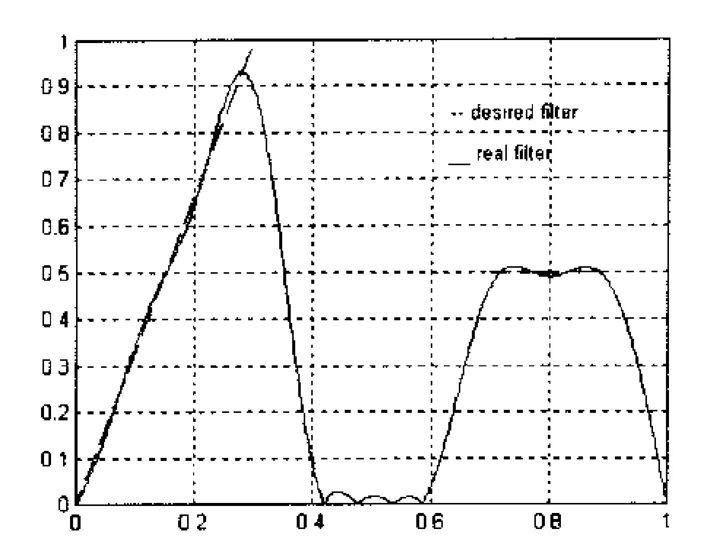


图 2.32　理想和实际分段带通滤波器特性

4. intfilt

功能：内插 FIR 滤波器设计。

格式：

b=intfilt(r, l, alpha)

b=intfilt(r, n, 'Lagrange')

说明：

b=intfilt(r, l, alpha)可设计出线性相位的 FIR 滤波器，它特别适用于对每 r 个样本交替出现 r−1 个零点的序列进行滤波，利用最近的 2l−1 个非零样本并假定原来的带宽为 $\alpha * f_n$(Nyquist 频率)时，滤波器可实现理想的限带内插。所设计的滤波器非常适合于 interp 函数采用。

b=intfilt(r, n, 'Lagrange')或 b=intfilt(r, n, 'l')设计的 FIR 滤波器，可对每 r 个样本交替出现 r−1 个零点的序列进行 n 阶拉格朗日多项式内插。

这两种滤波器均为基本的低通滤波器。

参见：interp，decimate，resample

5. remez

功能：Parks-McClellan 最优 FIR 滤波器设计。

格式：

b=remez(n, f, m)

b=remez(n, f, m, w)

b=remez(n, f, m, 'ftype')

b=remez(n, f, m, w, 'ftype')

说明：

remez 函数设计出采用 Parks-McClellan 算法的线性相位 FIR 滤波器，Parks-McClellan 算法使用了 Remez 交换算法和 Chebyshev 逼近理论，以设计出期望和实际频率响应之间最优拟合的滤波器，这种滤波器是在使期望频率响应和实际频率响应之间的最大误差最小的情况下最优。以这种方式设计的滤波器在频率响应上显现出等波纹，因此有时也称为等波纹滤波器。

b=remez(n，f，m)可得到 n 阶 FIR 滤波器，其幅频特性与由 f 和 m 指定的性能匹配。有关 f 和 m 的说明与 firls 函数中的 f 和 m 相同。

滤波器系数矢量 b 满足对称关系

$$b(k) = b(n+2-k) \quad k = 1, \cdots, n+1$$

b=remez(n，f，m，w)可利用权值矢量 w 对各频率段进行加权拟合，w 的长度为 f 和 m 长度的一半，用以指定各频率段的权值。

b=remez(n，f，m，'ftype')和 b=remez(n，f，m，w，'ftype')可指定滤波器的类型：

· 当 ftype=hilbert 时，设计的滤波器为奇对称的线性相位滤波器（Ⅲ型和Ⅳ型），b 的系数满足 b(k)=−b(n+2−k)，k=1，2，…，n+1。这类滤波器包括 Hilbert（希尔伯特）变换器。

· 当 ftype=differentiator 时，也设计出Ⅲ型和Ⅳ型滤波器，但采用了特殊的加权技术，对非零幅值的频段，将误差乘以系数 1/f，这样可使低频段误差大大小于高频段误差。

例 设计一 17 阶的 Parks-McClellan 带通滤波器，并画出期望幅频曲线和实际幅频曲线。其程序如下

```
f=[0  0.3  0.4  0.6  0.7  1];
m=[0  0  1  1  0  0];
b=remez(17, f, m);
[h, w]=freqz(b, 1, 512);
plot(f, m, w/pi, abs(h))
```

这可得到如图 2.33 所示的幅频曲线。

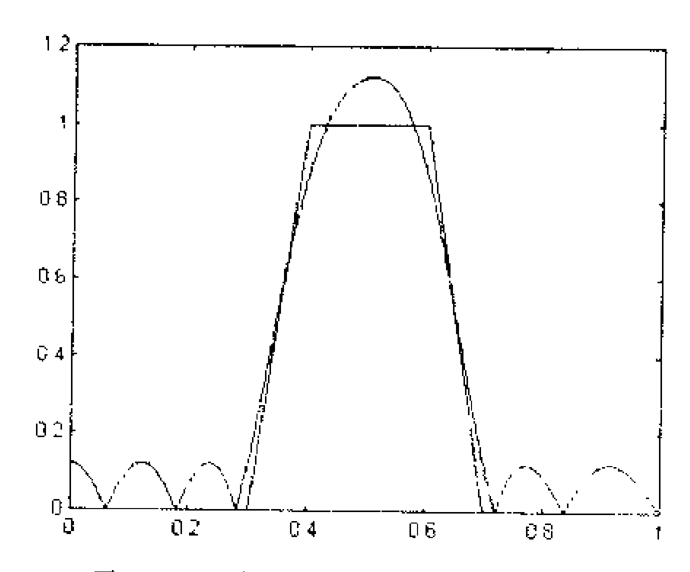


图 2.33 理想和实际带通滤波器幅频曲线

参见：butter，cheby1，cheby2，ellip，fir1，fir2，firls，remezord，yulewalk

6. remezord

功能：Parks-McClellan 最优 FIR 滤波器阶估计。

格式：

```
[n, fo, mo, w]=remezord(f, m, dev)
[n, fo, mo, w]=remezord(f, m, dev, Fs)
```

说明：

remezord 函数可为 remez 函数选择滤波器的阶。给定频域中的性能指标，remezord 可产生近似满足指标的最小阶。

[n，fo，mo，w]=remezord(f，m，dev)可找出近似的阶 n、归一化频带边界 fo、频带内幅值 mo 及加权矢量 w，使由 remez 函数构成的滤波器满足 f、m 及 dev 指定的性能要求。其中 f 和 m 用来指示频段及相应的幅值，这与 firls 函数类似，但应注意一点，f 中省去了 0 和 1 这两点，即

```
length(f)=2*length(m)-2
```

dev 用于指定各频段允许的偏差。

[n，fo，mo，w]=remezord(f，m，dev，Fs)可指定取样频率 Fs，如果缺省 Fs，则默认为 2 Hz。

应该注意，有时 remezord 函数对阶估计不足，因此可能会使滤波器达不到指定的性能，这时应稍微增加阶次。

例 设计一最小阶的低通滤波器，通带截止频率为 500 Hz，阻带截止频率为 600 Hz(取样频率为 2000 Hz)，通带波纹小于 3 dB，阻带低于 40 dB。其程序如下

```
Rp=3; Rs=40;
Fs=2000; f=[500 600];
m=[1 0];
dev=[(10^(Rp/20)-1)/(10^(Rp/20)+1)  10^(-Rs/20)];
[n, fo, mo, w]=remezord(f, m, dev, Fs);
b=remez(n, fo, mo, w);
[h, f]=freqz(b, 1, 1024, Fs);
plot(f, 20*log10(abs(h)))
```

这时得到如图 2.34 所示的低通滤波器特性。

由图 2.34 可以看出，滤波器性能还有点不满足指定的性能，这时可使滤波器阶次增加 1 阶，其程序如下

```
n=n+1;
b=remez(n, fo, mo, w);
[h, f]=freqz(b, 1, 1024, Fs);
plot(f, 20*log10(abs(h)))
```

这时可得到如图 2.35 所示的低通滤波器特性。

参见：buttord，cheb1ord，cheb2ord，ellipord，remez

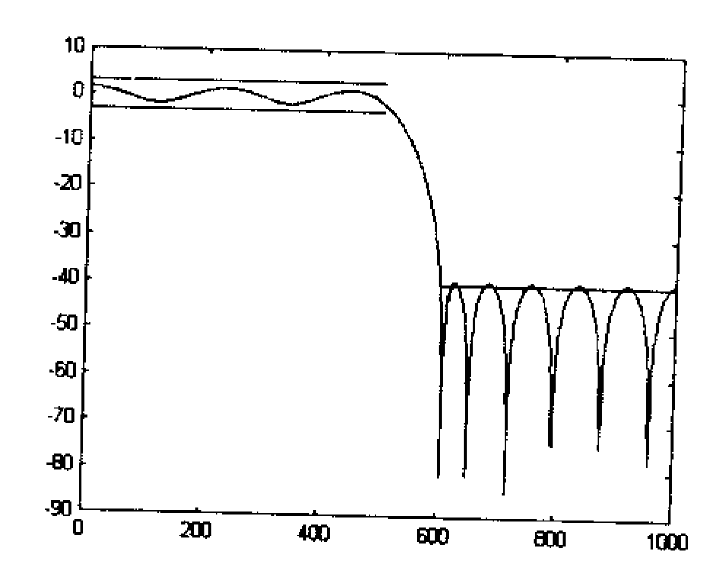


图 2.34 低通滤波器特性(n=22)

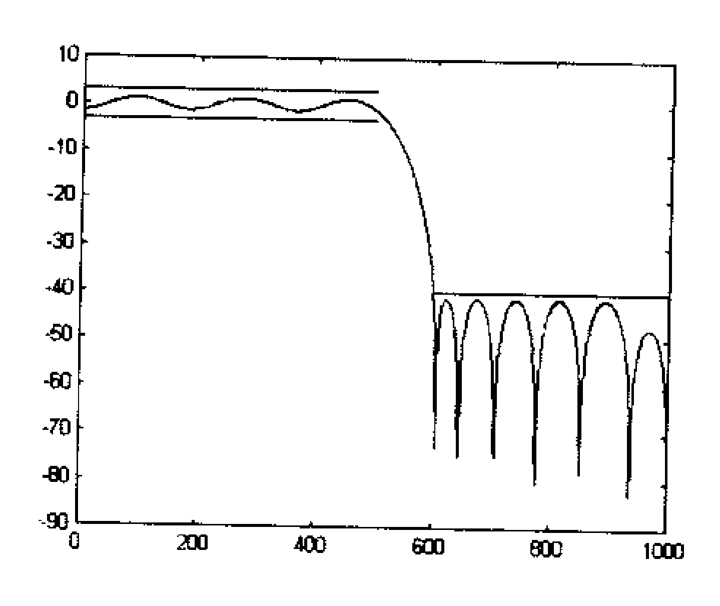


图 2.35 低通滤波器特性(n=23)

2.7 变　　换

1. czt

功能：线性调频 Z 变换。

格式：

```
y=czt(x, m, w, a)
y=czt(x)
```

说明：

y=czt(x，m，w，a)可计算 x 信号的线性调频 Z 变换，它计算 x 沿着由 w 和 a 定义的螺旋周线上的 Z 变换，m 指定变换长度，w 指定沿着 z 平面螺旋周线上的点之间的比率(即倾斜率)，a 指定起始点。

当 x 为矩阵时，czt(x，m，w，a)对 x 的列进行变换。

y=czt(x)直接采用默认值：m=length(x)，w=exp(j * 2 * pi/m)，a=1，这实际上就等效于 DFT 或 FFT。

参见： fft，freqz

2．dct

功能： 离散余弦变换(DCT)。

格式：

y=dct(x)

y=dct(x，n)

说明：

y=dct(x)可完成对 x 的离散余弦变换：

$$y(k) = \sum_{i=1}^{n} 2x(i)\cos\left[\frac{\pi}{2n}k(2i+1)\right] \qquad k=0,1,\cdots,n-1$$

其中 n 为矢量 x 的长度。当 x 为矩阵时，dct 函数按 x 的列进行变换。

y=dct(x，n)，在变换之前将 x 的长度补足或截断至 n。

参见： fft，idct

3．idct

功能： 逆离散余弦变换。

格式：

x=idct(y)

x=idct(y，n)

说明：

逆离散余弦变换是从它的离散余弦变换(DCT)系数中重构一序列，idct 函数为 dct 的逆函数。

x=idct(y)可求出 y 的逆离散余弦变换：

$$x(i) = \frac{1}{n}\sum_{k=0}^{n-1} w(k)y(k)\cos\frac{\pi}{2n}k(2i+1)$$

其中

$$w(k) = \begin{cases} \frac{1}{2} & k=0 \\ 1 & 1 \leqslant k \leqslant n-1 \end{cases}$$

n=length(x)

x=idct(y，n)在变换前将 y 的长度补足或截断至 n。

参见： dct，ifft

4. dftmtx

功能：离散傅里叶变换矩阵。

格式：

```
A=dftmtx(n)
```

说明：

离散傅里叶变换矩阵是单位圆附近的复矩阵，它与一矢量的乘积可给出该矢量的离散傅里叶变换。

A=dftmtx(n)得一 n×n 矩阵 A，这时若另有一 n 维列矢量 x，则

```
y=A*x
```

可得到 x 的离散傅里叶变换。

逆离散傅里叶变换矩阵为

```
Ai=conj(dftmtx(n))/n
```

参见：convmtx，fft

5. fft

功能：一维快速傅里叶变换(FFT)。

格式：

```
y=fft(x)
y=fft(x, n)
```

说明：

fft 函数用于计算矢量或矩阵的离散傅里叶变换，这可通过

$$X(k+1)=\sum_{n=0}^{N-1}x(n+1)W_N^{kn}$$

实现，其中 N=length(x)，$W_N=e^{-j(2\pi/N)}$。

y=fft(x)为利用 FFT 算法计算矢量 x 的离散傅里叶变换，当 x 为矩阵时，y 为矩阵 x 每一列的 FFT。当 x 的长度为 2 的幂次方时，则 fft 函数采用基 2 的 FFT 算法，否则采用稍慢的混合基算法。

y=fft(x，n)采用 n 点 FFT。当 x 的长度小于 n 时，fft 函数在 x 的尾部补零，以构成 n 点数据；当 x 的长度大于 n 时，fft 函数会截断序列 x。当 x 为矩阵时，fft 函数按类似的方式处理列长度。

例 考虑一被噪声污染的信号，很难看出它所包含的频率分量，如一个由 50 Hz 和 120 Hz 正弦信号构成的信号，受零均值随机噪声的干扰，数据采样率为 1000 Hz。现可通过 fft 函数来分析其信号频率成分。其程序如下

```
t=0:0.001:0.6;
x=sin(2*pi*50*t)+sin(2*pi*120*t);
y=x+1.5*randn(1, length(t));
Y=fft(y, 512);
```

```
P=Y. * conj(Y)/512;      %计算功率谱密度
f=1000 * (0:255)/512;
plot(f, p(1:256))
```

这样可得到如图 2.36 所示的信号功率谱密度。从图中可以推测，信号集中在 120 Hz 和 50 Hz。

参见：dct，dftmtx，fft2，fftshift，filter，freqz，ifft

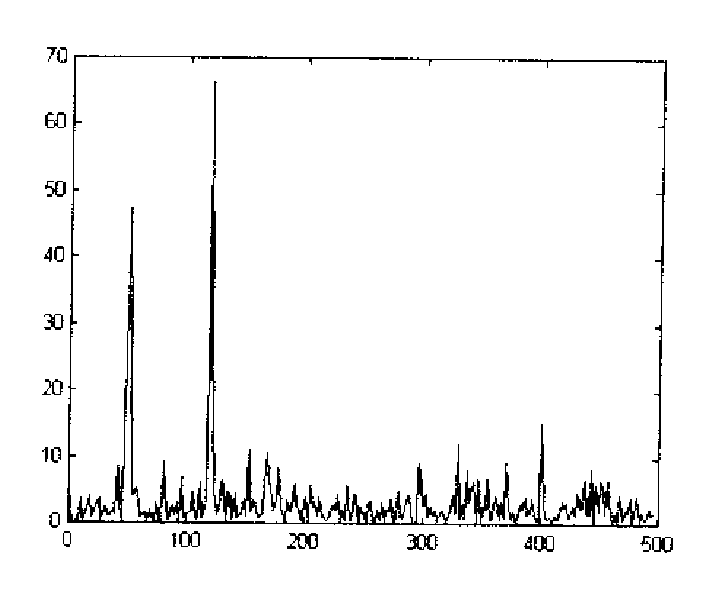


图 2.36 信号功率谱密度

6. ifft

功能：一维逆快速傅里叶变换(IFFT)。

格式：

```
y=ifft(x)
y=ifft(x, n)
```

说明：

ifft 函数用于计算矢量或矩阵的逆傅里叶变换，即

$$x(n+1)=\frac{1}{N}\sum_{k=0}^{N-1}X(k+1)W_N^{-kn}$$

其中 N=length(x)，$W_N=e^{-j(2\pi/N)}$。

y=ifft(x)用于计算矢量 x 的 IFFT。当 x 为矩阵时，计算所得的 y 为矩阵 x 中每一列的 IFFT。

y=ifft(x, n)采用 n 点 IFFT。当 length(x)<n 时，在 x 中补零；当 length(x)>n 时，将 x 截断，使 lgenth(x)=n。

ifft 函数是 fft 函数的逆，其相应的 M 文件中采用的算法类似。

参见：fft，fft2，fftshift，ifft2

7. fftshift

功能：重新排列 fft 和 fft2 的输出。

格式：

```
y=fftshift(x)
```

说明：

y=fftshift(x)可重排 fft 和 fft2 函数产生的结果，即将零频分量移到频谱的中心，这在实际应用时是很方便的。

当 x 为矢量时，fftshift(x)直接将 x 中的左右两半交换而产生 y。

当 x 为矩阵时，fftshift(x)将 x 的 4 个四分之一分块两两交换。

例

```
    x=[0.5  1; -0.3  0.4];
    y=fft2(x)
y=
     1.6000     -1.2000
     1.4000      0.2000
   z=fftshift(Y)
z=
       0.2000     1.4000
     -1.2000      1.6000
```

参见：fft，fft2

8. hilbert

功能：Hilbert(希尔伯特)变换。

格式：

```
y=hilbert(x)
```

说明：

y=hilbert(x)可从实数据序列中得到有时称作“解析信号”的复螺旋线序列。解析信号由实部和虚部构成，其实部为原来的序列，虚部为其 Hilbert 变换，虚部由实部加上 90°相移构成。由于正弦变换成余弦，余弦变换成正弦，因此 Hilbert 变换后的数据与原始数据具有相同的幅值和频率范围，并且包含了原始数据中的相位信息。

当 x 为矩阵时，则 y=hilbert(x)按列进行变换。

Hilbert 变换对计算时序列的瞬时特性，特别是幅度和频率特性是非常有用的。瞬时幅度是复 Hilbert 变换的幅度，瞬时频率是瞬时相角的时间变化率。对一纯正弦信号，瞬时幅度和频率均为常量，然而瞬时相角为锯齿形，这反映出它在一个周期内的局部线性变化。对混合正弦信号，这些特性是几个点的局部平均。

参见：fft，ifft，rceps

2.8 统计信号处理

1. cov

/**功能**：协方差矩阵。

格式：

c=cov(x)

c=cov(x, y)

说明：

cov 函数用于计算协方差矩阵。当 x 为矢量时，c=cov(x)可求出矢量 x 的方差 c(标量)，当 x 为矩阵时，x 的每一行为一观察值，每一列为一变量，这样 cov(x)就得到协方差矩阵。diag(cov(x))则为由每列的方差所构成的矢量，sqrt(diag(cov(x))即为标准差矢量。

c=cov(x, y)中，x, y 为长度相等的列矢量，这样 c=cov(x, y)就相当于 c=cov([x y])。

cov 函数在计算之前，在每列中去除其均值。

参见：corrcoef, xcorr, xcov

mean, median, std(参见楼顺天等编著. MATLAB 程序设计语言. 西安电子科技大学出版社，1997)

2. xcov

功能：互协方差函数估计。

格式：

v=xcov(x)

v=xcov(x, y)

v=xcov(x, 'option')

v=xcov(x, y, 'option')

说明：

xcov 函数用于计算随机过程的互协方差序列，在特殊情况下，可计算自协方差。

互协方差是去除均值后的互相关序列：

$$\varphi_{xy}(m) = E\{(x_n - m_x)(y_{n+m}^* - m_y)\}$$

其中 m_x 和 m_y 分别是 x 和 y 序列的均值。xcov 函数只是估计序列的互协方差，这是因为实际上我们只能得到无限长随机过程的有限个时间段。

v=xcov(x, y)格式中，x 和 y 是长度为 M 的矢量，则函数可计算出长度为 2M−1 的互协方差矢量。

v=xcov(x)中，当 x 为矢量时，可计算出自协方差序列。当 x 为 M×P 的矩阵时，v=xcov(x)可得到(2M−1)×P^2 的互协方差阵。

在缺省情况下，xcov 函数计算非归一化的行协方差序列，即

$$C_{nxy}(m) = \sum_{n=0}^{N-|m'|-1} x(n)y^*(n+m)$$

但为了合适地估计相关函数，需要进行归一化处理。xcov(x，'option')和 xcov(x，y，'option')中的 option 选项可用来指定计算方式：

- 当 option=biased 时，计算互相关函数的有偏估计；
- 当 option=unbiased 时，计算互相关函数的无偏估计；
- 当 option=coeff 时，对序列进行归一化，使零滞后的自相关函数为 1.0；
- 当 option=none 时，即为缺省情况。

参见：corrcoef，cov，xcorr，xcorr2，conv

3. corrcoef

功能：相关系数矩阵。

格式：

c=corrcoef(x)

c=corrcoef(x，y)

说明：

corrcoef 函数可从输入矩阵中计算出相关系数矩阵。如果记 c=cov(x)，则函数 corrcoef(x)可写成

$$\text{corrcoef}(i, j)=\frac{c(i, j)}{\sqrt{c(i, i)c(j, j)}}$$

即　c=corrcoef(x)为零滞后的协方差函数；

c=corrcoef(x，y)等同于 corrcoef([x y])函数。

参见：cov，xcorr，xcov

4. xcorr

功能：互相关函数估计。

格式：

c=xcorr(x，y)

c=xcorr(x)

c=xcorr(x，'option')

c=xcorr(x，y，'option')

说明：

xcorr 函数可估计出随机过程的互相关序列，在特殊情况下可得到自相关序列。

对平稳随机过程 x_n 和 y_n，其实际的互相关序列为

$$\gamma_{xy}(m) = E\{x_n y_{n+m}^*\}$$

由于我们只能得到有限个数据样本，因此 xcorr 函数只能估计出 $\gamma_{xy}(m)$。

当 x，y 矢量的长度为 M 时，c=xocrr(x，y)可得到长度为 2M+1 的互相关序列。而 c=xcorr(x)(x 为矢量时)可计算出矢量 x 的自相关序列；当 x 为 M×P 矩阵时，c=xcorr(x)

可得到(2M－1)×P^2的互相关矩阵。

在缺省 option 时，xcorr 函数计算非归一化的行相关：

$$c_{nxy}(m) = \sum_{n=0}^{N-|m|-1} x(n)y^*(n+m)$$

但为了正确估计相关函数，应该进行归一化处理。xcorr(x，'option')和 xcorr(x，y，'option')中的 option 可用来指定相关选项：

- 当 option＝biased 时，xcorr 函数可计算互相关函数的有偏估计，即

$$c_{nxy}(m) = \frac{1}{N} \sum_{n=0}^{N-|m|-1} x(n)y^*(n+m)$$

- 当 option＝unbiased 时，xcorr 函数可计算互相关函数的无偏估计，即

$$c_{nxy}(m) = \frac{1}{N-|m|} \sum_{n=0}^{N-|m|-1} x(n)y^*(n+m)$$

- 当 option＝coeff 时，xcorr 函数对序列进行归一化处理，这样零滞后的自相关序列恒为 1.0。
- 当 option＝none 时，即为缺省情况。

例 输入

```
x=1:3;
y=4:6;
c=xcorr(x, y)
```

可得结果

```
c=
   12.000    23.000    32.000  17.000    6.000
```

参见：corrcoef，conv，cov，xcorr2，xcov

5. cohere

功能：估计两信号间相关函数平方的幅值。

格式：

```
Cxy=cohere(x)
Cxy=cohere(x, y, nfft)
[Cxy, f]=cohere(x, y, nfft, Fs)
Cxy=cohere(x, y, nfft, Fs, window)
Cxy=cohere(x, y, nfft, Fs, window, noverlap)

Cxy=cohere(x, y, …, 'dflag')
cohere(x, y)
```

说明：

Cxy＝cohere(x，y)可计算出等长度矢量 x 和 y 之间相关性的幅度平方，它是 x、y 的功率谱及互功率谱的函数，即

$$C_{xy}(f)=\frac{|P_{xy}(f)|^2}{P_{xx}(f)P_{yy}(f)}$$

Cxy=cohere(x，y)采用了以下的缺省值：

- nfft=min(256，length(x))
- Fs=2
- window=hanning(nfft)
- noverlap=0

nfft 指定了 FFT 的长度，其值决定了要进行相关估计的频率值；Fs 为取样频率；window 指定了加窗函数；noverlap 指定分段重叠的样本数。在命令格式中省略某个参数，就取其缺省值。

当 x 和 y 为实数时，cohere 函数只估计正频率处的相关函数。这时，当 nfft 为偶数时，输出 Cxy 为 nfft/2+1 维的列矢量；当 nfft 为奇数时，Cxy 为(nfft+1)/2 维的列矢量。当 x 或 y 为复数时，cohere 函数估计正负频率处的相关函数，Cxy 的长度为 nfft。

Cxy=cohere(x，y，nfft)可在估计 x 的功率谱中采用指定长度的 FFT，当 nfft 取为 2 的幂次方时，可提高执行速度。

[Cxy，f]=cohere(x，y，nfft，Fs)可在 f 矢量得到估计相关性的频率点，Fs 为取样频率，它只影响频率轴刻度，对输出 Cxy 没有影响。

Cxy=cohere(x，y，nfft，Fs，window)可指定加窗函数，当 window 为标量时，说明采用缺省的 Hanning(汉宁窗)，其长度为 window 值，窗长度必须小于等于 nfft。

Cxy=cohere(x，y，nfft，Fs，window，noverlap)使 x 矢量分段之间重叠 noverlap 个取样值。

除 x、y 参数外，都可用空矩阵来指定采用缺省值，例如：

```
Cxy=cohere(x, y, [ ], [ ], kaiser(128, 5))
```

则使用 nfft=256，Fs=2。

Cxy=cohere(x，y，…，'dflag')可利用 dflag 指定处理方式：

- 当 dflag=linear 时，可从预加权的 x、y 段中删去最佳的直线拟合；
- 当 dflag=mean 时，可从预加权的 x、y 段中删去均值；
- 当 dflag=none 时，不作任何处理。

不带输出变量的 cohere 函数可在当前图形窗口中绘出相关估计的频域曲线。

例 绘出两有色噪声之间的相关估计。其程序如下

```
h=fir1(30, .2, boxcar(31));      %设计低通滤波器
h1=ones(1, 10)/sqrt(10);
r=randn(16384, 1);               %产生白噪声
x=filter(h1, 1, r);              %产生有色噪声
y=filter(h, 1, x);               %产生有色噪声
cohere(x, y, 1024, [ ], [ ], 512)
```

结果如图 2.37 所示。

参见：csd，psd，tfe

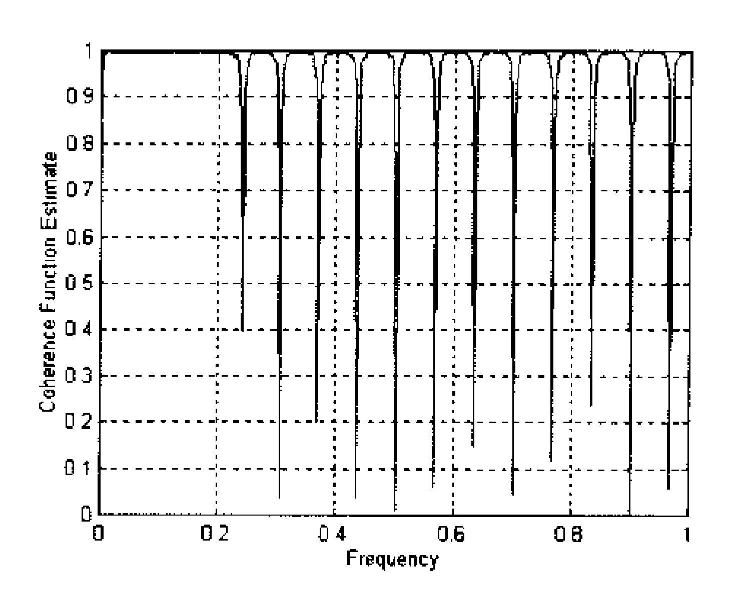


图 2.37 相关函数估计

6. csd

功能：估计两信号间的互谱密度(CSD)。

格式：

```
Pxy=csd(x)
Pxy=csd(x, y, nfft)
[Pxy, f]=csd(x, y, nfft, Fs)
Pxy=csd(x, y, nfft, Fs, window)
Pxy=csd(x, y, nfft, Fs, window, noverlap)

Pxy=csd(x, y, …, 'dflag')
csd(x, y)
[Pxy, Pxyc, f]=csd(x, y, nfft, Fs, window, noverlap, p)
```

说明：

Pxy=csd(x, y)可计算出等长度矢量 x 和 y 之间的互谱密度(CSD)，它与 cohere 函数非常类似，只是两者计算的结果不同。除最后一种格式外，其余 7 种格式的说明与 cohere 函数完全相同，nfft 用于指定所采用的 FFT 的长度，Fs 指定取样频率，window 指定窗函数，noverlap 指定分段之间的重叠取样数，详见 cohere 函数。

最后一种格式：[Pxy, Pxyc, f]=csd(x, y, nfft, Fs, window, noverlap, p)可得到 Pxy 的 p * 100%置信区间的估计 Pxyc，其中 p∈[0, 1)，这样区间[Pxy－Pxyc, Pxy＋Pxyc]就以概念 p 覆盖了真值 CSD。plot(f, [Pxy Pxy－Pxyc Pxy＋Pxyc])可绘出 p * 100%置信区间内的互频谱，如未指定 p，则取缺省值 0.95。

例 产生两个有色噪声，并绘出具有 95%置信区间的 CSD。其程序如下

```
h=fir1(30, .2, boxcar(31));
```

```
h1=ones(1, 10)/sqrt(10);
r=randn(16384, 1);
x=filter(h1, 1, r);
y=filter(h, 1, x);
csd(x, y, 1024, 10000, triang(500), 0, [ ])
```

结果如图 2.38 所示。

参见：cohere，psd，tfe

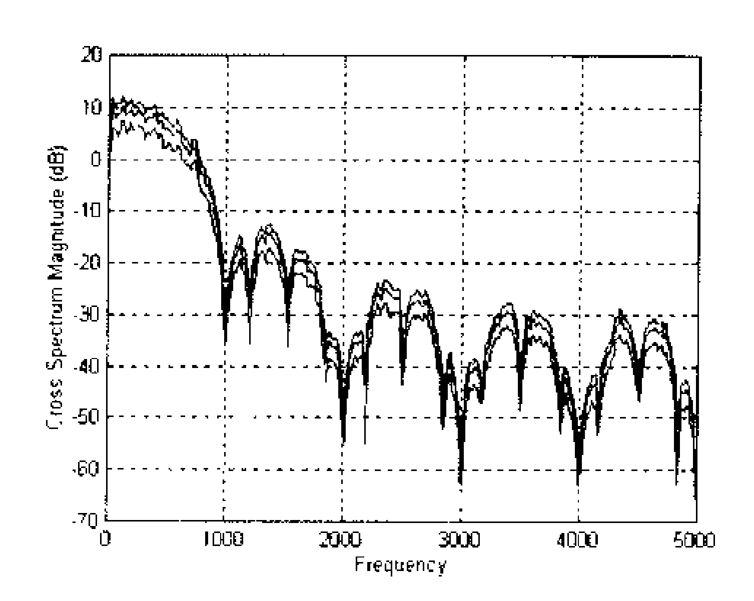


图 2.38　互谱密度(CSD)曲线

7. psd

功能：估计信号的功率谱密度(PSD)。

格式：

```
Pxx=psd(x)
Pxx=psd(x, nfft)
[Pxx, f]=psd(x, nfft, Fs)
Pxx=psd(x, nfft, Fs, window)
Pxx=psd(x, nfft, Fs, window, noverlap)

Pxx=psd(x, …, 'dflag')
psd(x)
[Pxx, Pxxc, f]=psd(x, nfft, Fs, window, noverlap, p)
```

说明：

Pxx=psd(x)估计序列 x 的功率谱密度。psd 函数与 cohere 函数非常类似，只是两者计算结果不同。除最后一种格式外，其余格式的说明与 cohere 函数完全相同，nfft 用于指定所采用的 FFT 的长度，Fs 指定取样频率，window 指定窗函数，noverlap 指定分段之间的

重叠取样数，详见 cohere 函数。

最后一种格式：[Pxx，Pxxc，f]=psd(x，nfft，Fs，window，noverlap，p)可得到 Pxx 的 p＊100%置信区间的估计 Pxxc，其中 p∈[0，1]，这样区间[Pxx－Pxxc，Pxx＋Pxxc]就以概念 p 覆盖了真值 PSD。plot(f，[Pxx　Pxx－Pxxc　Pxx＋Pxxc])可绘出 p＊100%置信区间内的功率谱。如未指定 p，则取缺省值 0.95。

例　产生一有色噪声，并绘出具有 95%置信区间的 PSD。其程序如下

```
h=fir1(30，.2，boxcar(31))；
r=randn(16384，1)；
x=filter(h1，1，r)；
psd(x，1024，10000，kaiser(512，5)，0，.95)
```

这样就得到了如图 2.39 所示的功率谱密度曲线。

参见：cohere，csd，tfe

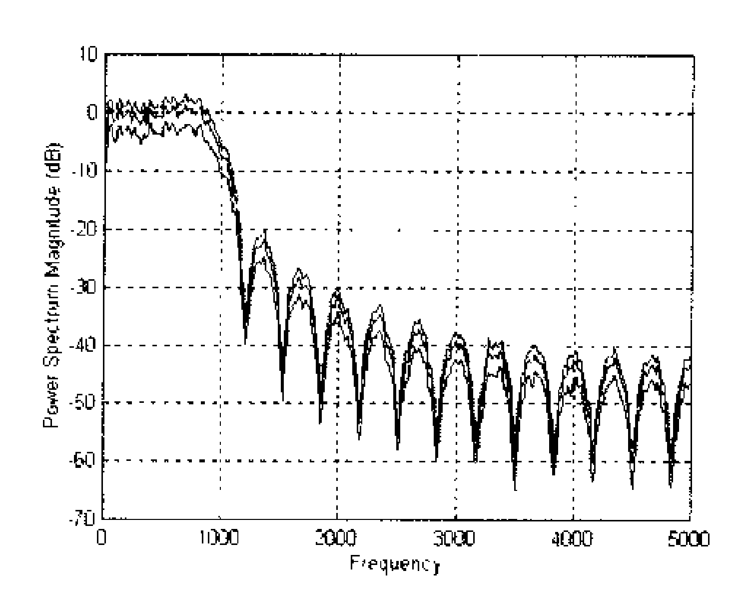


图 2.39　功率谱密度曲线

8. tfe

功能：从输入输出中估计传递函数。

格式：

```
Txy=tfe(x，y)
Txy=tfe(x，y，nfft)
[Txy，f]=tfe(x，y，nfft，Fs)
Txy=tfe(x，y，nfft，Fs，window)
Txy=tfe(x，y，nfft，Fs，window，noverlap)

Txy=tfe(x，y，…，'dflag')
tfe(x，y)
```

说明：

Txy＝tfe(x，y)可从输入输出中估计出传递函数，它是 x 和 y 的互谱密度与 x 的功率谱的商，其公式为

$$T_{xy}(f) = \frac{P_{xy}(f)}{P_{xx}(f)}$$

从函数计算上看，tfe 函数与 chohere 函数完全一样，因此有关 tfe 函数各种格式的说明可参见 cohere 函数。

例 计算两有色噪声序列间的传递函数，并绘出传递函数的特性。其程序如下

```
h=fir1(30, .2, boxcar(31));
x=randn(16384, 1);
y=filter(h, 1, x);
tfe(x, y, 1024, [ ], [ ], 512)
```

这样就得到如图 2.40 所示的传递函数的特性曲线。

参见：cohere，csd，psd

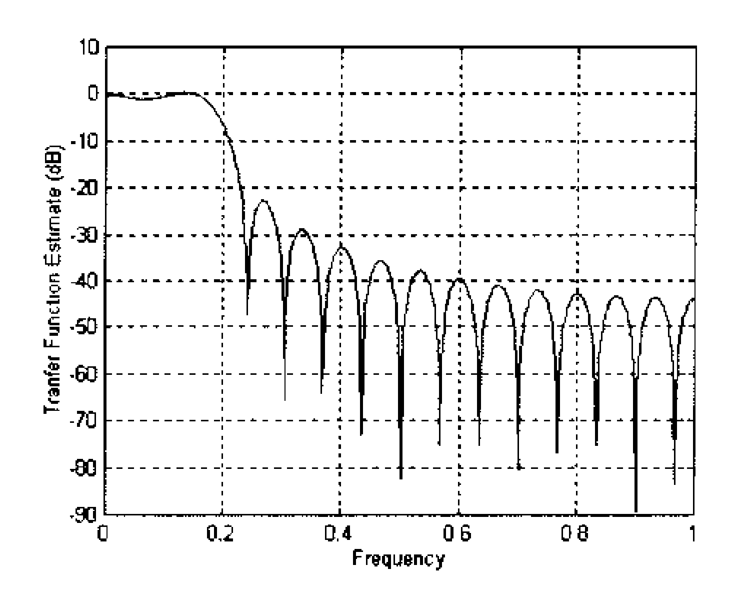


图 2.40 传递函数特性曲线

2.9 窗 函 数

1. boxcar

功能：矩形窗。

格式：

w＝boxcar(n)

说明：

boxcar(n)函数可产生一长度为 n 的矩形窗函数。提供这一函数的目的是使窗函数部分

内容完整。实际上对一序列加上矩形窗，就等同于没有加窗。

参见：bartlett，blackman，chebwin，hamming，hanning，kaiser，triang

2. triang

功能：三角窗。

格式：

```
w=triang(n)
```

说明：

triang(n)函数可得到 n 点的三角窗函数。三角窗系数为

当 n 为奇数时：

$$w(k)=\begin{cases}\dfrac{2k}{n+1} & 1\leqslant k\leqslant \dfrac{n+1}{2}\\[2ex] \dfrac{2(n-k+1)}{n+1} & \dfrac{n+1}{2}\leqslant k\leqslant n\end{cases}$$

当 n 为偶数时：

$$w(k)=\begin{cases}\dfrac{2k-1}{n} & 1\leqslant k\leqslant \dfrac{n}{2}\\[2ex] \dfrac{2(n-k+1)}{n} & \dfrac{n}{2}\leqslant k\leqslant n\end{cases}$$

三角窗函数非常类似于 Bartlett 窗。Bartlett 在取样点 1 和 n 上总是以零结束，而三角窗在这些点上并不为零。

实际上，当 n 为奇数时，triang(n−2)的中心 n−2 个点等效于 bartlett(n)。

参见：bartlett，blackman，boxcar，chebwin，hamming，hanning，kaiser

3. bartlett

功能：Bartlett(巴特利特)窗。

格式：

```
w=bartlett(n)
```

说明：

bartlett(n)可得到 n 点 Bartlett 窗函数。Bartlett 窗函数系数为

$$w(k)=\begin{cases}\dfrac{2(k-1)}{n-1} & 1\leqslant k\leqslant \dfrac{n+1}{2}\\[2ex] 2-\dfrac{2(k-1)}{n-1} & \dfrac{n+1}{2}\leqslant k\leqslant n\end{cases}$$

Bartlett 函数与三角窗函数非常类似。当 n 为奇数时，bartlett(n)的中心 n−2 点等效于 triang(n−2)。

参见：blackman，boxcar，chebwin，hamming，hanning，kaiser，triang

4. hamming

功能：Hamming(哈明)窗。

格式：

w=hamming(n)

说明：

hamming(n)可产生 n 点的 Hamming 窗，其系数为

$$w(k+1) = 0.54 - 0.46\cos\left(2\pi\frac{k}{n-1}\right) \qquad k = 0, \cdots, n-1$$

参见： bartlett，blackman，boxcar，chebwin，hanning，kaiser，triang

5. hanning

功能： Hanning(汉宁)窗。

格式：

w=hanning(n)

说明：

hanning(n)可产生 n 点的 Hanning 窗，其系数为

$$w(k) = 0.5\left[1 - \cos\left(2\pi\frac{k}{n+1}\right)\right] \quad k = 1, \cdots, n$$

参见： bartlett，blackman，boxcar，chebwin，hamming，kaiser，triang

6. blackman

功能： Blackman(布莱克曼)窗。

格式：

w=blackman(n)

说明：

blackman(n)可产生 n 点的 Blackman 窗，其系数为

$$w(k) = 0.42 - 0.5\cos\left(2\pi\frac{k-1}{n-1}\right) + 0.8\cos\left(4\pi\frac{k-1}{n-1}\right) \quad k = 1, \cdots, n$$

与等长度的 Hamming 和 Hanning 窗相比，Blackman 窗的主瓣稍宽，旁瓣稍低。

参见： bartlett，boxcar，chebwin，hamming，hanning，kaiser，triang

7. chebwin

功能： Chebyshev(切比雪夫)窗。

格式：

w=chebwin(n，r)

说明：

w=chebwin(n，r)可产生 n 点的 Chebyshev 窗函数，其傅里叶变换后的旁瓣波纹低于主瓣 r dB。注意当 n 为偶数时，窗函数的长度为 n+1。

参见： bartlett，blackman，boxcar，hamming，hanning，kaiser，triang

8. kaiser

功能：Kaiser(凯泽)窗。

格式：

```
w=kaiser(n, beta)
```

说明：

w=kaiser(n, beta)可产生 n 点的 Kaiser 窗函数，其中 beta 为影响窗函数旁瓣的 β 参数，其最小的旁瓣抑止 α 与 β 之间的关系为

$$\beta=\begin{cases}0.1102(\alpha-0.87) & \alpha>50\\ 0.5842(\alpha-21)^{0.4}+0.07886(\alpha-21) & 21\leqslant\alpha\leqslant 50\\ 0 & \alpha<21\end{cases}$$

增加 β 可使主瓣变宽，旁瓣的幅度降低。

参见：bartlett, blackman, boxcar, chebwin, hamming, hanning, triang

2.10 参数化建模

1. invfreqs

功能：从频域数据中辨识连续时间滤波器。

格式：

```
[b, a]=invfreqs(h, w, nb, na)
[b, a]=invfreqs(h, w, nb, na, wt)
[b, a]=invfreqs(h, w, nb, na, wt, iter)
[b, a]=invfreqs(h, w, nb, na, wt, iter, tol)
[b, a]=invfreqs(h, w, nb, na, wt, iter, tol, 'trace')
```

说明：

invfreqs 函数是 freqs 的逆操作，它可找出拟合给定复频响应的一连续时间传递函数。从实验分析观点，invfreqs 函数可用于将幅值和相位数据转换成传递函数。

在[b, a]=invfreqs(h, w, nb, na)中，h 和 w 用于指定频率幅度响应及相应的频率，得到的 b，a 可构成滤波器

$$H(s)=\frac{B(s)}{A(s)}=\frac{b(1)s^{n_b}+b(2)s^{(n_b-1)}+\cdots+b(n_b+1)}{a(1)s^{n_a}+a(2)s^{(n_a-1)}+\cdots+a(n_a+1)}$$

其中 n_b、n_a 分别指定分子和分母多项式的阶次。

w 中的频率以弧度为单位，length(w)=length(h)。

[b, a]=invfreqs(h, w, nb, na, wt)可对拟合误差进行加权，wt 为加权矢量，且 length(wt)=length(w)。

[b, a]=invfreqs(h, w, nb, na, wt, iter, tol)可利用数值迭代的方法寻找最优拟合，

从而确保所得线性系统的稳定性。参数 iter 告诉 invfreqs 函数当解收敛时，结束迭代过程，或者在 iter 次迭代后一旦满足 tol 容限，就结束迭代。这里的收敛与 tol 有关，当梯度矢量范数小于 tol 时，就认为收敛。tol 缺省时，默认为 0.01。

如果采用全 1 加权阵，只须用空阵代替，即 invfreqs(h，w，nb，na，[ ]，iter，tol)。

如要得到一有关迭代过程的报告，可采用

```
[b, a]=invfreqs(h, w, nb, na, wt, iter, tol, 'trace')
```

例 先利用 freqs 函数产生一已知系统的频率响应[h，w]，然后利用 invfreqs 函数重构这一系统，从而可与原系统进行对比。输入

```
a=[1  2  3  2  1  4];
b=[1  2  3  2  3];
[h, w]=freqs(b, a, 64);
[bb, aa]=invfreqs(h, w, 4, 5)
bb=
   1.0000    2.0000    3.0000    2.0000    3.0000
aa=
   1.0000    2.0000    3.0000    2.0000    1.0000    4.0000
```

这与原系统是完全一致的。然而 aa 中含有左半平面的极点，因而系统不稳定。但利用 invfreqs函数的迭代算法可找出一逼近系统的稳定方案：

```
[bbb, aaa]=invfreqs(h, w, 4, 5, [ ], 30)
bbb=
   0.6602    2.0058    2.3589    1.0837    -0.1337
aaa=
   1.0000    3.2986    7.0223    5.7543    2.9129    0.0002
```

参见：freqs，prony，invfreqz，freqz

2. invfreqz

功能：从频域数据中辨识离散时间滤波器。

格式：

```
[b, a]=invfreqz(h, w, nb, na)
[b, a]=invfreqz(h, w, nb, na, wt)
[b, a]=invfreqz(h, w, nb, na, wt, iter)
[b, a]=invfreqz(h, w, nb, na, wt, iter, tol)
[b, a]=invfreqz(h, w, nb, na, wt, iter, tol, 'trace')
```

说明：

invfreqz 函数是 freqz 的逆操作，它可找出拟合给定频率响应的一离散时间传递函数。从实验分析的观点，invfreqz 函数对将幅值和相位数据转换成对应的传递函数是很有用的。

invfreqz 函数与 invfreqs 函数几乎一样，其说明可参见 invfreqs 函数的说明。

例 输入

```
a=[1  2  3  2  1  4];
```

```
b=[1  2  3  2  3];
[h, w]=freqz(b, a, 64);
[bb, aa]=invfreqz(h, w, 4, 5)
bb=
  1.0000    2.0000    3.0000    2.0000    3.0000
aa=
  1.0000    2.0000    3.0000    2.0000    1.0000    4.0000
```

为得到稳定的线性系统，输入

```
[bbb, aaa]=invfreqz(h, w, 4, 5, [ ], 30)
bbb=
  0.2427    0.2788    0.0069    0.0971    0.1980
aaa=
  1.0000   -0.8944    0.6954    0.9997   -0.8933    0.6949
```

参见：freqz，prony，invfreqs，freqs

3. prony

功能：时间域 IIR 滤波器设计的 Prony 方法。

格式：

```
[b, a]=prony(h, nb, na)
```

说明：

Prony 方法是一种在已知时域冲激响应时构造 IIR 滤波器的算法，它可应用于滤波器设计、指数信号建模和系统辨识(参数化建模)。

[b，a]=prony(h，nb，na)可找出一滤波器，其传递函数的分于阶次为 nb，分母阶次为 na，且时域冲激响应为 h。其滤波器可表示为

$$H(z)=\frac{B(z)}{A(z)}=\frac{b(1)+b(2)z^{-1}+\cdots+b(n_b+1)z^{-n_b}}{1+a(2)z^{-1}+\cdots+a(n_a+1)z^{-n_a}}$$

例 先得到 Butterworth 滤波器的冲激响应，然后再从这恢复出滤波器的系数。输入

```
[b, a]=butter(4, .2)
b=
  0.0048    0.0193    0.0289    0.0193    0.0048
a=
  1.0000   -2.3659    2.3140   -1.0547    0.1874
h=filter(b, a, [1 zeros(1, 25)]);
[bb, aa]=prony(h, 4, 4)
bb=
  0.0048    0.0193    0.0289    0.0193    0.0048
aa=
  1.0000   -2.3659    2.3140   -1.0547    0.1874
```

参见：butter，cheby1，cheby2，ellip，invfreqz，levinson，lpc，stmcb

4. stmcb

功能：利用 Steiglitz-McBride 迭代方法求线性模型。

格式：

```
[b, a]=stmcb(h, nb, na)
[b, a]=stmcb(h, nb, na, iter)
[h, a]=stmcb(h, nb, na, iter, ai)
[b, a]=stmcb(x, u, nb, na)
```

说明：

Steiglitz-McBride 迭代是在给定时域冲激响应时确定 IIR 滤波器的一种算法，它已在滤波器设计和系统辨识中得到应用。

[b, a]=stmcb(h, nb, na)中，h 指定时域冲激响应，nb 和 na 分别是传递函数分子和分母多项式的阶次。

[b, a]=stmcb(h, nb, na, iter)中，可指定迭代次数 iter，缺省时取为 5。

[b, a]=stmcb(h, nb, na, iter, ai)中，可指定分母多项式系数估值的初值 ai。

[b, a]=stmcb(x, u, nb, na)中，u 和 x 为系统相应的输入和输出矢量，其长度相等。当然在这种格式中，也可以使用 iter 和 ai 选项。

例 输入

```
[b, a]=butter(6, .2);
h=filter(b, a, [1, zeros(1, 100)]);
freqz(b, a, 128)
```

这可以得到原 Butterworth 滤波器的特性曲线，如图 2.41 所示。

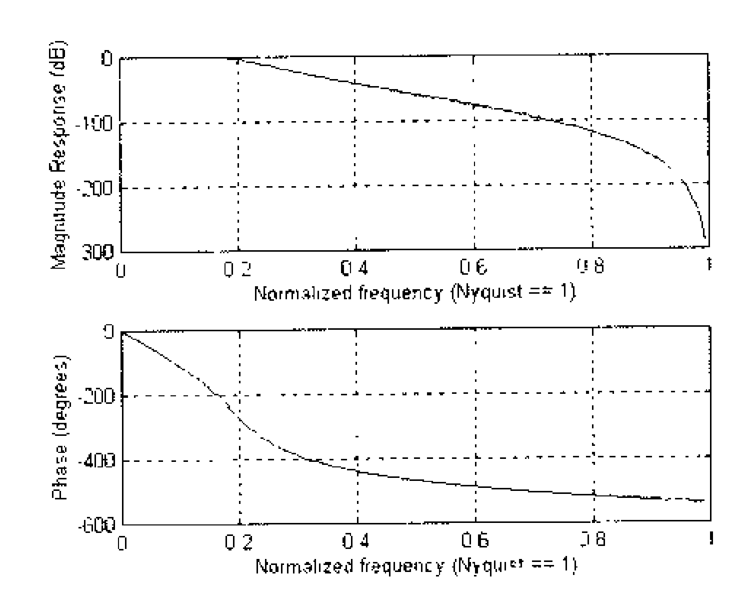


图 2.41 Butterworth 滤波器特性曲线

继续输入

```
[bb, aa]=stmcb(h, 4, 4);
freqz(bb, aa, 128)
```

可得到构造的滤波器特性，如图 2.42 所示。

参见：lpc，prony

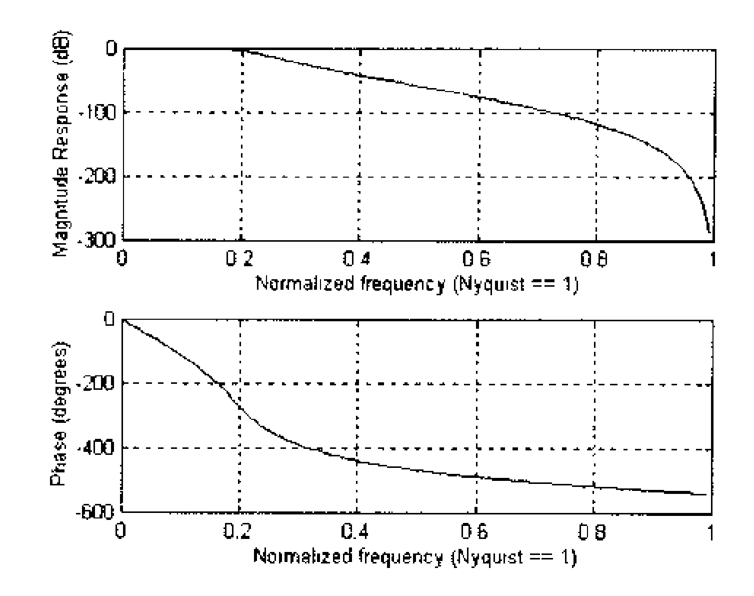


图 2.42 滤波器特性曲线

5. levinson

功能：Levinson-Durbin 递归算法。

格式：

```
a=levinson(r, n)
```

说明：

Levinson-Durbin 递归是给定确定性的自相关序列时确定全极点型 IIR 滤波器的一种算法，它可应用于滤波器设计、解码和谱估计中，由 Levinson-Durbin 递归算法得到的滤波器是最小相位滤波器。

a=levinson(r, n)可得到 n 阶自回归线性过程的系数 a，其中 r 指定这一过程的自相关序列，所产生的滤波器为

$$H(z)=\frac{1}{A(z)}=\frac{1}{1+a(2)z^{-1}+\cdots+a(n+1)z^{-n}}$$

参见：lpc，prony，stmcb

6. lpc

功能：线性预测系数。

格式：

```
a=lpc(x, n)
```

说明：

线性预测指将每一取样信号看成是以前信号取样的线性组合，即为全极点 IIR 滤波器的输出。它可应用于滤波器设计、语言编码、谱分析及系统辨识。

a=lpc(x，n)可得到 n 阶自回归线性模型的系数，即

$$x(k) = -a(2)x(k-1) - a(3)x(k-2) - \cdots - a(n+1)x(k-n-1)$$

其中 x 为实输入时序列，n 为分母多项式 A(z)的阶。

例 利用 lpc 函数建立具有全极点 IIR 滤波器特性的非递归 FIR 滤波器。其程序如下

```
x=[1:4  4:-1:1];
a=lpc(x, 15);
[h, w]=freqz(x, 1, 512);
[h1, w]=freqz(1, a, 512);
plot(w/pi, abs(h), w/pi, abs(h1), '--')
```

这样就得到如图 2.43 所示的曲线，其中实线表示 FIR 滤波器响应，而虚线表示 IIR 滤波器响应。

参见：levinson，prony，stmcb

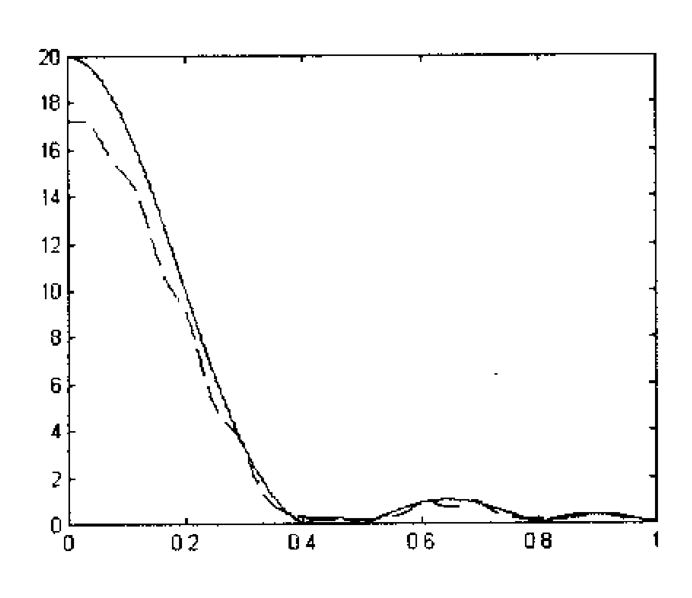


图 2.43 FIR 和 IIR 幅频特性

2.11 特殊操作

1．rceps

功能：实倒谱和最小相位重构。

格式：

```
y=rceps(x)
[y, ym]=rceps(x)
```

说明：

实倒谱是傅里叶变换序列幅度实对数的逆傅里叶变换。rceps(x)函数可得到实序列 x 的实倒谱，实倒谱是实值函数。

[y, ym]=rceps(x)可产生输入序列 x 的实倒谱 y 和最小相位重构 ym。

参见：cceps，fft，hilbert，unwrap

2. cceps

功能：倒谱分析和最小相位重构。

格式：

```
y=cceps(x)
```

说明：

倒谱分析是非线性信号处理技术，它与语音信号处理、同态滤波等密切相关。

y=cceps(x)可得到实序列 x 的复倒谱，即为傅里叶变换序列复对数的逆傅里叶变换，复倒谱为实值函数。

参见：rceps，fft，hilbert，unwrap

3. decimate

功能：降低序列的取样速率。

格式：

```
y=decimate(x, r)
y=decimate(x, r, n)
y=decimate(x, r, 'fir')
y=decimate(x, r, n, 'fir')
```

说明：

decimate 可使序列的取样率降低，与内插过程相反，它将输入数据经低通滤波器滤波后，得到的平滑信号再以较低的取样率重新取样。

y=decimate(x, r)可将 x 的取样率降至原取样率的 1/r，因此 y 的长度为 x 的 1/r。在缺省情况下，decimate 函数采用 8 阶低通 Chebyshev Ⅰ型滤波器进行滤波，它以前向和反向对输入数据进行滤波，以消除相位失真，并使滤波器的阶次加倍。

y=decimate(x, r, n)采用 n 阶的 Chebyshev 滤波器，从数值稳定角度出来，不宜采用 13 阶以上的滤波器。

y=decimate(x, r, 'fir')采用 30 点的 FIR 滤波器，这时按单方向对输入数据进行滤波。

y=decimate(x, r, n, 'fir')采用 n 点 FIR 滤波器。

例 利用 decimate 函数使信号的取样率降至原取样率的 1/4，并利用 stem 函数绘出信号。

```
t=0:0.00025:1;
x=sin(2*pi*30*t)+sin(2*pi*60*t);
```

```
y=decimate(x, 4);
subplot(2, 1, 1)
stem(x(1:120))
subplot(2, 1, 2)
stem(y(1:30))
```

这样就得到了如图 2.44 所示的信号图。

参见：interp，resample

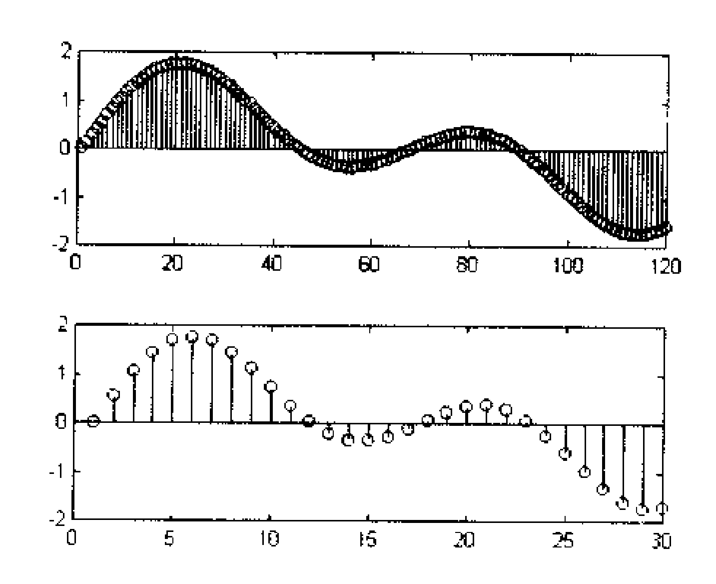


图 2.44　信号图

4. interp

功能：提高取样速率(内插)。

格式：

```
y=interp(x, r)
y=interp(x, r, l, alpha)
[y, b]=interp(x, r, l, alpha)
```

说明：

通过内插可提高序列的取样速率，interp 函数利用一特殊的低通滤波器，从而实现低通内插。

y=interp(x, r)可使 x 序列的取样率增加 r 倍，因此 y 的长度为 x 长度的 r 倍。

y=interp(x, r, l, alpha)可指定滤波器的长度 l 及其截止频率 alpha，其缺省值分别为 4 和 0.5。

[y, b]=interp(x, r, l, alpha)还产生了用于内插的低通滤波器的系数 b。

例　以 4 倍率内插一输入序列，这时输入

```
t=0:.001:1;
```

```
x=sin(2*pi*30*t)+sin(2*pi*60*t);
y=interp(x, 4);
subplot(2, 1, 1)
stem(x(1:30))
subplot(2, 1, 2)
stem(y(1:120))
```

这时得到了如图 2.45 所示的信号图。

参见：decimate，resample

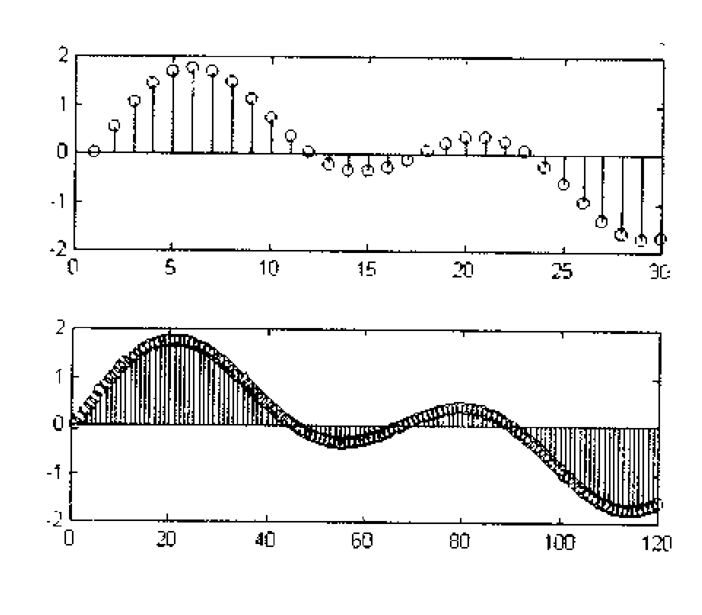


图 2.45　信号图

5. resample

功能：改变取样速率。

格式：

```
y=resample(x, p, q)
y=resample(x, p, q, n)
y=resample(x, p, q, n, beta)
y=resample(x, p, q, b)
[y, b]=resample(x, p, q)
```

说明：

y=resample(x，p，q)函数利用多相实现方法，使 x 序列的取样率变为(p/q)f_0(f_0 为原来 x 的取样率)，因此 y 的长度应为(p/q)length(x)。

在对序列 x 进行重新取样的过程中，resample 函数采用了一种抗混淆(低通)FIR 滤波器，这种滤波器为带 kaiser 窗的滤波器，它可由 fir1 函数设计。

y=resample(x，p，q，n)可利用 x[n]两边的 n 项来构成重新取样，resample 所使用的滤波器长度正比于 n。n 越大精度越高，但计算时间也更长。n 缺省时取为 10，当 n=0 时，

表示 reqsample 采用零阶保持器。

y=resample(x, p, q, n, beta)中，beta 用作 kaiser 窗函数的设计参数，beta 的缺省值为 5。

y=resample(x, p, q, b)，利用指定的滤波器 b 对 x 进行滤波。

[y, b]=resample(x, p, q)除得到 y 外，还可得到在重新取样过程中所采用的滤波器系数 b。

参见：decimate, fir1, interp, intfilt, kaiser

6. medfilt1

功能：一维中值滤波。

格式：

y=medfilt1(x, n)

y=medfilt1(x, n, blksz)

说明：

y=medfilt1(x, n)可得到输入序列 x 通过一 n 阶中值滤波器后的输出。当 n 为奇数时，y(k)为 $x\left(k-\frac{n-1}{2}:k+\frac{n-1}{2}\right)$ 的中值；当 n 为偶数时，y(k)为 $x\left(k-\frac{n}{2}:k+\frac{n}{2}-1\right)$ 的中值。如果 n 缺省，则取为 3。

在 y=medfilt1(x, n, blksz)格式中，可利用 blksz 参数指定每次循环所计算的输出数，这在内存欠缺的情况下是很有用的。

当 x 为矩阵时，medfilt1 函数对 x 的列数据进行中值滤波，即

y(:, i)=medfilt1(x(:, i), n, blksz)

参见：filter, median

7. deconv

功能：反卷积和多项式除法。

格式：

[q, r]=deconv(b, a)

说明：

[q, r]=deconv(b, a)函数，采用长除法，从矢量 b 中反卷积矢量 a，结果商放在 q 矢量中，余数保存在 r 矢量中，即

b=conv(q, a)+r

当 a, b 为多项式系数时，a, b 的卷积等效于多项式乘法，反卷积等效于多项式除法。deconv 与 conv 函数是相互对应的函数。

参见：conv, filter, residuez

8. modulate

功能：通讯仿真中的调制。

格式：

y=modulate(x, Fc, Fs, 'method')

y=modulate(x, Fc, Fs, 'method', opt)

[y, t]=modulate(x, Fc, Fs)

说明：

在 y=modulate(x, Fc, Fs, 'method')中，x 为实信号，该函数对 x 信号进行调制，载波频率为 Fc，取样频率为 Fs，method 为所采用的调制方法，y 为调制后的信号。method 可取：

am 或 amdsb-sc	幅度调制，双边带，抑制载波。这时输出为 y = x. * cos(2 * pi * Fc * t)
amdsb-tc	幅度调制，双边带，传输载波。这时输出为 y=(x—opt). * cos(2 * pi * Fc * t) opt 的缺省值为 min(min(x))
amssb	幅度调制，单边带。这时输出为 y=x. * cos(2 * pi * Fc * t)+… +imag(hilbert(x)). * sin(2 * pi * Fc * t)
fm	频率调制。这时输出为 y=cos(2 * pi * Fc * t+opt * cumsum(x)) 其中 cumsum 是 x 积分的矩形逼近，opt 的缺省值为 opt=(Fc/Fs) * 2 * pi/(max(max(x))
pm	相位调制。这时输出为 y=cos(2 * pi * Fc * t+opt * x) 这时 opt 的缺省值为 opt=pi/(max(max(x)))
pwm	脉宽调制。这时从输入 x 中构成脉宽调制信号，其 x 的值(0～1)用于指定脉宽
ptm	脉冲时间调制。这时从 x 中构成脉冲时间调制信号，其 x 的值(0～1)用于确定脉冲的起始时刻
qam	正交幅度调制。其输出为 y=x. * cos(2 * pi * Fc * t)+opt. * sin(2 * pi * Fc * t) 注意参数 opt 与 x 等长

当 method 缺省时，则取 am。一般情况下，y 与 x 等长，但 pwm 和 ptm 调制时，y 与 x 不等长。

当 x 为矩阵时，modulate 对其列元素进行调制。

为计算调制信号，modulate 函数要建立一内部的时间矢量 t，这一矢量可以由下列命令得到

[y, t]=modulate(x, Fc, Fs)

参见：demod，vco

9. demod

功能：通讯仿真中的解调。

格式：

```
x=demod(y, Fc, Fs, 'method')
y=demod(y, Fc, Fs, 'method', opt)
[x1, x2]=demod(y, Fc, Fs, 'qam')
```

说明：

demod 函数完成对信号的解调，即从调制信号中恢复原来的信号。modulate 与 demod 函数构成了通信仿真中所必须的调制和解调功能。

x=demod(y, Fc, Fs, 'method', opt)函数利用载波频率 Fc，取样频率 Fs 对实载波信号 y 进行解调，其可用的方法由 method 指定，有关说明详见 modulate 函数。

参见：modulate，vco

10. vco

功能：电压控制振荡器。

格式：

```
y=vco(x, Fc, Fs)
y=vco(x, [Fmin Fmax], Fs)
```

说明：

y=voc(x, Fc, Fs)可产生一取样频率为 Fs 的振荡信号，其振荡频率由实输入矢量或矩阵 x 确定。Fc 为载波或参考频率，x∈[−1, 1]，当 x=−1 时，输出 y 的频率为 0；当 x=0时，输出 y 的频率为 Fc；当 x=1 时，输出 y 的频率为 2*Fc。其波形形状均为余弦。

y=vco(x, [Fmin Fmax], Fs)可调整频率调制范围，使 x=−1 时产生 Fmin 的振荡输出，x=1 时产生 Fmax 的振荡输出。当然 Fmin 和 Fmax 应小于 Fs/2。

Fs 的缺省值为 1，Fc 的缺省值为 Fs/4。

当 x 为矩阵时，vco 函数根据 x 的列产生调制信号。

例 利用 vco 函数产生瞬时频率为时间三角函数的信号，其采样频率为 5000 Hz，然后利用 specgram 函数绘出信号的频谱图。其程序如下

```
Fs=5000;
t=0:1/Fs:2;
x=vco(sawtooth(2*pi*t, .75), [.1 .4]*Fs, Fs);
specgram(x, 512, Fs, kaiser(256, 5), 220)
```

这可得到如图 2.46 所示的频谱图。

参见：demod，modulate

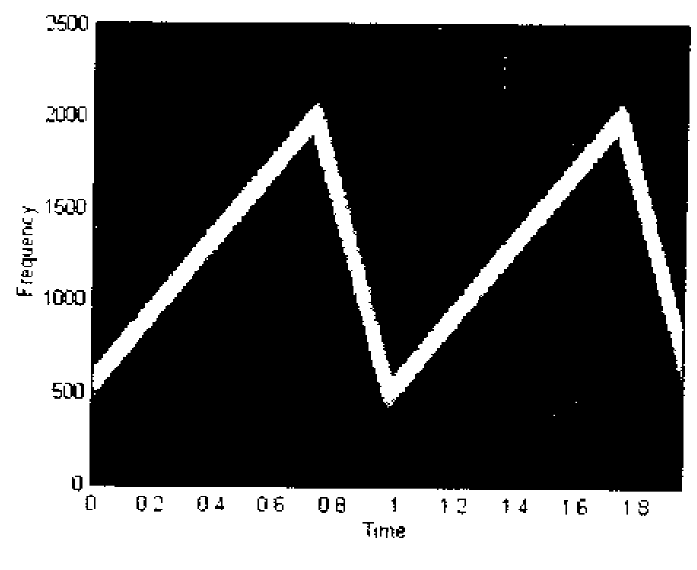


图 2.46 频谱图

11. specgram

功能：与时间有关的频率分析(频谱)。

格式：

B=specgram(a)

B=specgram(a, nfft)

[B, f]=specgram(a, nfft, Fs)

[B, f, t]=specgram(a, nfft, Fs)

B=specgram(a, nfft, Fs, window)

B=specgram(a, nfft, Fs, window, noverlap)

specgram(a)

说明：

specgram 函数利用一滑窗计算信号的加窗离散时间傅里叶变换，频谱就是这一函数的幅值。

B=specgram(a)计算出信号矢量 a 的频谱，它采用了一些缺省值：

- nfft=min(256, length(a))
- Fs=2
- window=hanning(nfft)
- noverlap=length(window)/2

nfft 指定 specgram 函数所采用的 FFT 的点数，它决定了计算离散傅里叶变换的频率点，Fs 为取样频率，window 指定窗函数，noverlap 指定分段重迭的点数。

当 a 为实数时，specgram 函数只在正频域内计算离散傅里叶变换，其长度 n 为偶数时，specgram 可产生 nfft/2+1 行；当 n 为奇数时，specgram 可产生 nfft/2 行。B 的列数为

k=fix((n−noverlap)/(length(window)−noverlap))

当 a 为复数时，specgram 函数要计算正负频域的离散傅里叶变换，这时 B 为 nfft 行的复矩阵，从 B 中的第 1 列第 1 点开始，时间随列线性增加，而频率从 0 开始随行线性下降。

B=specgram(a, nfft)，在计算中采用 nfft 点的 FFT，利用 2 的幂次方的 FFT 可加快计算速度。

[B, f]=specgram(a, nfft, Fs)除得到频谱 B 之外，还得到了计算离散傅里叶变换的频率矢量 f。这里 Fs 并不影响结果 B，但它改变了频率刻度。

[B, f, t]=specgram(a, nfft, Fs)，还得到了记录窗函数与 a 相交时间的矢量 t。

B = specgram(a, nfft, Fs, window)，可指定所采用的窗函数及 x 分段的点数。当 window为标量时，表示采用长度为 window 的 Hanning 窗，窗的长度应不大于 nfft，如果 nfft大于窗函数的长度，则自动将窗函数缺少部分添零补足 nfft。

B=specgram(a, nfft, Fs, window, noverlap)，将 x 的各段之间重叠 noverlap 点。

利用空矩阵[ ]可指定选项的缺省值，如

```
B=specgram(x, [ ], 10000)
```

等效于

```
B=specgram(x, min(256, length(x)), 10000)
```

不带输出变量的 specgram 函数可显示出频谱图。

有关示例可参见 vco 函数。

参见：cohere, csd, psd, tfe

2.12 模拟原型滤波器设计

1. besselap

功能：Bessel(贝塞尔)模拟低通滤波器原型。

格式：

```
[z, p, k]=besselap(n)
```

说明：

[z, p, k]=besselap(n)可设计出 n 阶 Bessel 模拟低通滤波器，其零点、极点及增益分别存放在 z, p 和 k 中。由于这种滤波器没有零点，因此 z 为空矩阵，且至多含有 25 个极点，即

$$H(s) = \frac{k}{(s-p(1))(s-p(2))\cdots(s-p(n))}$$

Bessel 原型在低频和高频段渐近等效于同阶的 Butterworth 原型。模拟 Bessel 滤波器可由群延迟来表征，它在零频处最平坦，在通带内几乎为常数。零频处的群延迟为

$$\left(\frac{(2n)!}{2^n n!}\right)^{\frac{1}{n}}$$

参见：besself, buttap, cheb1ap, cheb2ap, ellipap

2．buttap

功能：Butterworth（比特沃思）模拟低通滤波器原型。

格式：

[z，p，k]=buttap(n)

说明：

[z，p，k]=buttap(n)可设计出 n 阶 Butterworth 模拟低通滤波器原型，其传递函数为

$$H(s)=\frac{k}{(s-p(1))(s-p(2))\cdots(s-p(n))}$$

因此实际上 z 为空阵。Butterworth 滤波器可由通带内最平坦、总体上单调的幅度特性来表征。

在低通滤波器中，Butterworth 幅度响应平方的前 2n－1 阶导数在 ω=0 处为零，幅度响应的平方为

$$|H(\omega)|^2=\frac{1}{1+(\omega/\omega_0)^{2n}}$$

参见：butter，besselap，cheb1ap，cheb2ap，ellipap

3．cheb1ap

功能：Chebyshev（切比雪夫）Ⅰ型模拟低通滤波器原型。

格式：

[z，p，k]=cheb1ap(n，Rp)

说明：

[z，p，k]=cheb1ap(n，Rp)可设计出 Chebyshev Ⅰ型模拟低通滤波器原型，其通带内的波纹系数为 Rp 分贝，传递函数为

$$H(s)=\frac{k}{(s-p(1))(s-p(2))\cdots(s-p(n))}$$

Chebyshev Ⅰ型滤波器为通带内等波纹、阻带内单调的滤波器，其极点均匀分布在左半平面内的某椭圆上。

参见：cheby1，buttap，besselap，cheb2ap，ellipap

4．cheb2ap

功能：Chebyshev（切比雪夫）Ⅱ型模拟低通滤波器原型。

格式：

[z，p，k]=cheb2ap(n，Rs)

说明：

[z，p，k]=cheb2ap(n，Rs)可设计出 Chebyshev Ⅱ型模拟低通滤波器原型，其阻带内的波纹系数低于通带 Rs 分贝，传递函数为

$$H(s)=k\frac{(s-z(1))(s-z(2))\cdots(s-z(n))}{(s-p(1))(s-p(2))\cdots(s-p(n))}$$

Chebyshev Ⅱ型滤波器为通带内单调、阻带内等波纹的滤波器，其极点位置为cheb1ap中极点位置的倒数。

参见：cheby2，besselap，buttap，cheb1ap，ellipap

5．ellipap

功能：椭圆模拟低通滤波器原型。

格式：

[z，p，k]=ellipap(n，Rp，Rs)

说明：

[z，p，k]=ellipap(n，Rp，Rs)可设计出椭圆模拟低通滤波器原型，其通带波纹为Rp分贝，阻带的波纹为低于通带的Rs分贝。传递函数为

$$H(s) = k\frac{(s-z(1))(s-z(2))\cdots(s-z(n))}{(s-p(1))(s-p(2))\cdots(s-p(n))}$$

椭圆滤波器在通带和阻带内均为等波纹，它可提供比Butterworth和Chebyshev滤波器更陡的下降斜度，但会损失通带和阻带的波纹指标。

参见：ellip，besselap，buttap，cheb1ap，cheb2ap

2.13 频率变换

1．lp2bp

功能：低通到带通模拟滤波器变换。

格式：

[bt，at]=lp2bp(b，a，Wo，Bw)

[At，Bt，Ct，Dt]=lp2bp(A，B，C，D，Wo，Bw)

说明：

lp2bp函数可将截止频率为1弧度/秒的模拟低通滤波器原型变换成具有指定带宽Bw和中心频率Wo的带通滤波器，这种变换是利用butter、cheby1、cheby2、ellip函数设计数字滤波器中的一步。

lp2bp函数可以两种形式表示：传递函数和状态空间，但输入系统必须为模拟滤波器原型。

[bt，at]=lp2bp(b，a，Wo，Bw)可将传递函数表示的模拟低通滤波器原型转换成带通滤波器，其中心频率为Wo，带宽为Bw。模拟低通滤波器原型可表示为

$$H(s) = \frac{b(s)}{a(s)} = \frac{b(1)s^{nb} + \cdots + b(nb)s + b(nb+1)}{a(1)s^{na} + \cdots + a(na)s + a(na+1)}$$

如果要求滤波器的低端截止频率为w1，高端截止频率为w2，则可计算出Wo和Bw：

Wo=sqrt(w1 * w2)

Bw=w2−w1

[At, Bt, Ct, Dt]=lp2bp(A, B, C, D, Wo, Bw)可将以连续状态方程表示的低通滤波器原型变换成带通滤波器，其中心频率为 Wo、通带为 Bw。低通滤波器可表示成

$$\dot{x} = Ax + Bu$$
$$y = Cx + Du$$

参见：bilinear, lp2bs, lp2hp, lp2lp

2. lp2hp

功能：低通到高通模拟滤波器变换。

格式：

[bt, at]=lp2hp(b, a, Wo)

[At, Bt, Ct, Dt]=lp2hp(A, B, C, D, Wo)

说明：

lp2hp 函数将截止频率为 1 弧度/秒的模拟低通滤波器原型变换成截止频率为 Wo 的高通滤波器，这种变换是利用 butter、cheby1、cheby2、ellip 函数设计数字滤波器中的一步。

lp2hp 函数可有两种表示方法：传递函数和状态空间，但输入系统必须为模拟滤波器原型。

[bt, at]=lp2hp(b, a, Wo)可将传递函数表示的模拟低通滤波器原型变换成高通滤波器，其截止频率为 Wo。模拟低通滤波器原型可表示成

$$H(s) = \frac{b(s)}{a(s)} = \frac{b(1)s^{nb} + \cdots + b(nb)s + b(nb+1)}{a(1)s^{na} + \cdots + a(na)s + a(na+1)}$$

[At, Bt, Ct, Dt]=lp2hp(A, B, C, D, Wo)可将以连续状态空间表示的低通滤波器原型变换成高通滤波器，其截止频率为 Wo。低通滤波器可表示成

$$x = Ax + Bu$$
$$y = Cx + Du$$

参见：bilinear, lp2bp, lp2bs, lp2lp

3. lp2bs

功能：低通到带阻模拟滤波器变换。

格式：

[bt, at]=lp2bs(b, a, Wo, Bw)

[At, Bt, Ct, Dt]=lp2bs(A, B, C, D, Wo, Bw)

说明：

lp2bs 函数可将截止频率为 1 弧度/秒的模拟低通滤波器原型变换成具有指定带宽(Bw)和中心频率(Wo)的带阻滤波器，这种变换也是设计数字滤波器中的一步。

lp2bs 函数与 lp2bp 函数非常类似，只是由 lp2bs 函数设计出带阻滤波器，而由 lp2bp 函数设计出带通滤波器，其它说明与 lp2bp 函数相同，详见 lp2bp 函数。

参见：bilinear, lp2bp, lp2hp, lp2lp

4. lp2lp

功能：低通到低通模拟滤波器变换。

格式：

[bt, at]=lp2lp(b, a, Wo)

[At, Bt, Ct, Dt]=lp2lp(A, B, C, D, Wo)

说明：

lp2lp 函数可将截止频率为 1 弧度/秒的模拟低通滤波器原型变换成截止频率为 Wo 的低通滤波器，这种变换是利用 butter、cheby1、cheby2、ellip 函数设计数字滤波器中的一步。

lp2lp 函数可有两种表示形式：传递函数和状态空间，但输入系统必须为模拟滤波器原型。

[bt, at]=lp2lp(b, a, Wo)可将传递函数表示的模拟低通滤波器原型转换成低通滤波器，其截止频率为 Wo。模拟低通滤波器原型可表示成

$$H(s)=\frac{b(s)}{a(s)}=\frac{b(1)s^{nb}+\cdots+b(nb)s+b(nb+1)}{a(1)s^{na}+\cdots+a(na)s+a(na+1)}$$

[At, Bt, Ct, Dt]=lp2lp(A, B, C, D, Wo)可将以连续状态方程表示的低通滤波器原型变换成低通滤波器，其截止频率为 Wo。低通滤波器可表示成

$$\dot{x}=Ax+Bu$$
$$y=Cx+Du$$

参见：bilinear, lp2bp, lp2bs, lp2hp

2.14 滤波器离散化

1. bilinear

功能：双线性变换。

格式：

[zd, pd, kd]=bilinear(z, p, k, Fs)

[zd, pd, kd]=bilinear(z, p, k, Fs, Fp)

[numd, dend]=bilinear(num, den, Fs)

[numd, dend]=bilinear(num, den, Fs, Fp)

[Ad, Bd, Cd, Dd]=bilinear(A, B, C, D, Fs)

说明：

双线性变换为变量的映射关系，在数字滤波器中，它是将 s 域或模拟域映射成 z 域或数字域的标准方法，它可将以经典滤波器设计技术设计的模拟滤波器变换成离散等效滤波器。

双线性变换映射关系为

$$H(z) = H(s)|_{s=2f_s\frac{z-1}{z-1}}$$

这种变换可将 jΩ 轴($-\infty<\Omega<+\infty$)映射成单位圆 exp(jω)($-\pi\leqslant\omega\leqslant\pi$)，即

$$\omega = 2\tan^{-1}\left(\frac{\Omega}{2f_s}\right)$$

双线性变换函数 bilinear 可指定预卷绕的参数 Fp，这时变换关系为

$$H(z) = H(s)|_{s=\frac{2\pi f_p(z-1)}{\tan\left(\pi\frac{f_p}{f_s}\right)(z+1)}}$$

bilinear 函数可作用于 3 种不同的线性系统表示：零极点增益、传递函数、状态空间。

以零极点增益表示的线性系统为例，来说明 bilinear 函数，可表示为

```
[zd, pd, kd]=bilinear(z, p, k, Fs)
[zd, pd, kd]=bilinear(z, p, k, Fs, Fp)
```

其中 z，p，k 为 s 域传递函数的零点、极点和增益，Fs 为取样频率，zd，pd，kd 为经双线性变换后 z 域传递函数的零点、极点和增益，Fp 为预卷绕参数。

其它两种表示方法的格式类似。

参见：impinvar

2. impinvar

功能：冲激响应不变法实现模拟到数字的滤波器变换。

格式：

```
[bz, az]=impinvar(b, a, Fs)
[bz, az]=impinvar(b, a)
```

说明：

[bz，az]=impinvar(b，a，Fs)可将模拟滤波器(b，a)变换成数字滤波器(bz，az)，两者的冲激响应不变，即模拟滤波器的冲激响应按 Fs 取样后等同于数字滤波器的冲激响应。

[bz，az]=impinvar(b，a)采用 Fs 的缺省值 1 Hz。

例　将一模拟低通滤波器变换成数字滤波器(取样频率为 10 Hz)，利用 impinvar 可得到冲激响应相同的数字滤波器。

```
[b, a]=butter(4, .3, 's');
[bz, az]=impinvar(b, a, 10);
real(bz)
ans=
  1.0e-05 *
    0.0000      0.1324      0.5192      0.1273
real(az)
ans=
    1.0000     -3.9216      5.7679     -3.7709      0.9246
```

参见：bilinear，residuez

2.15 其　　它

1. conv2

功能：二维卷积。

格式：

```
C=conv2(A, B)
C=conv2(A, B, 'shape')
```

说明：

C=conv2(A，B)可计算矩阵A和B的二维卷积，如果把一个矩阵看作是二维FIR滤波器，则对另一个矩阵实现二维滤波。

设[ma，na]=size(A)，[mb，nb]=size(B)，则经C=conv2(A，B)后得到[ma+mb−1，na+nb−1]=size(C)。

C=conv2(A，B，'shape')可得到二维卷积变换结果的一部分，这由shape指定：

- 当shape=full时，得到整个二维卷积结果，这也就是缺省值；
- 当shape=same时，只得到卷积结果的中心部分，其大小为size(A)；
- 当shape=valid时，只得到未采用补0区域时的卷积结果；当size(A)>size(B)时，有size(C)=[ma−mb+1，na−nb+1]。

当size(A)>size(B)时，conv2执行速度最快。

参见：conv，deconv，filter2，xcorr，xcorr2

2. cplxpair

功能：将复数归成复共轭对。

格式：

```
y=cplxpair(x)
y=cplxpair(x, tol)
```

说明：

y=cplxpair(x)可将x的值按复共轭对形式重新排列，复共轭对按实部从小到大排列，再按虚部从小到大排列。最后列出所有的实数。

y=cplxpair(x，tol)设定一容限tol，以确定哪些数为实数。tol的缺省值为100 * eps。

例　均匀分布在单位圆上的5个点，可利用cplxpair归成复共轭对形式。

```
x=exp(2*pi*sqrt(-1)*(0:4)/5)';
y=cplxpair(x)
y=
   -0.8090   -0.5878i
   -0.8090   +0.5878i
```

```
0.3090   -0.9511i
0.3090   +0.9511i
1.0000
```

参见：无

3. detrend

功能：删除线性趋势。

格式：

y=detrend(x)

y=detrend(x, 0)

说明：

detrend 函数利用 FFT 处理算法，从 x 矢量或矩阵中删去均值或趋势。

y=detrend(x)可从矢量 x 中删去 x 的最优直线拟合，当 x 为矩阵，则按列进行处理。

y=detrend(x, 0)可从 x 中删去均值。

参见：polyfit（参见楼顺天等编著. MATLAB 程序设计语言. 西安电子科技大学出版社，1997）

4. fft2

功能：二维快速傅里叶变换。

格式：

Y=fft2(X)

Y=fft2(X, m, n)

说明：

Y=fft2(X)可完成矩阵 X 的二维 FFT，它是基于一维的 FFT 算法的。先按列对 X 中的每一列进行 FFT，然后对得到的结果（行向量）再作一次一维 FFT，从而得到了二维 FFT。

Y=fft2(X, m, n)中，指定对 X 截断或补零，使之成为 m×n 矩阵，然后再作二维的 FFT，因此其结果 Y 也为 m×n。

参见：fft, fftshift, ifft, ifft2

5. ifft2

功能：二维逆 FFT 变换。

格式：

Y=ifft2(X)

Y=ifft2(X, m, n)

说明：

Y=*ifft*2(X)可得到 X 矩阵的二维逆 FFT 变换。

Y=ifft2(X, m, n)在变换前，通过截断或补零使 X 阵成为 m×n 矩阵，然后再通过二

维逆 FFT 变换得到矩阵 Y。

ifft2 函数是 fft2 函数的逆函数。

参见：fft，fft2，fftshift，ifft

6. filter2

功能：二维数字滤波器。

格式：

Y=filter2(B，X)

Y=filter2(B，X，'shape')

说明：

Y=filter2(B，X)可利用由矩阵 B 表示的二维 FIR 滤波器，对输入数据矩阵 X 进行滤波，得到的输出 Y 与 X 同维。

Y=filter2(B，X，'shape')可通过 shape 选取滤波结果中的一部分作为输出。有关 shape 选项的说明参见 conv2 函数。

参见：conv2

7. polystab

功能：稳定多项式。

格式：

b=polystab(a)

说明：

polystab 函数可得到一稳定的多项式，它将幅值大于 1 的根映射到单位圆内。

b=polystab(a)中，a 通常为 z 域表示的多项式系数

$$a(z) = a(1) + a(2)z^{-1} + \cdots + a(na + 1)z^{-na}$$

经 polystab 函数后，可得到一稳定的多项式系数 b。

参见：roots

8. strips

功能：带状图。

格式：

strips(x)

strips(x，n)

strips(x，Sd，Fs)

说明：

strips(x)表示在长 250 的水平带状图上绘出矢量 x 的图形，当 x 为矩阵时，strips(x)绘出 x 中的每一列。

strips(x，n)表示在 n 个样本的带状图上绘出 x 的图形。

strips(x，Sd，Fs)表示在 Sd 秒的带状图上绘出 x 的图形(取样频率为 Fs)。

当 x 为复数时，则 strips 函数会忽略掉虚部。

例 在 0.25 秒的带状图上绘制出 2 秒的正弦频率调制信号。其程序如下

```
Fs=1000;
t=0:1/Fs:2;
x=vco(sin(2*pi*t), [10 490], Fs);
strips(x, .25, Fs)
```

这样可得到如图 2.47 所示的带状图。

参见：plot，stem

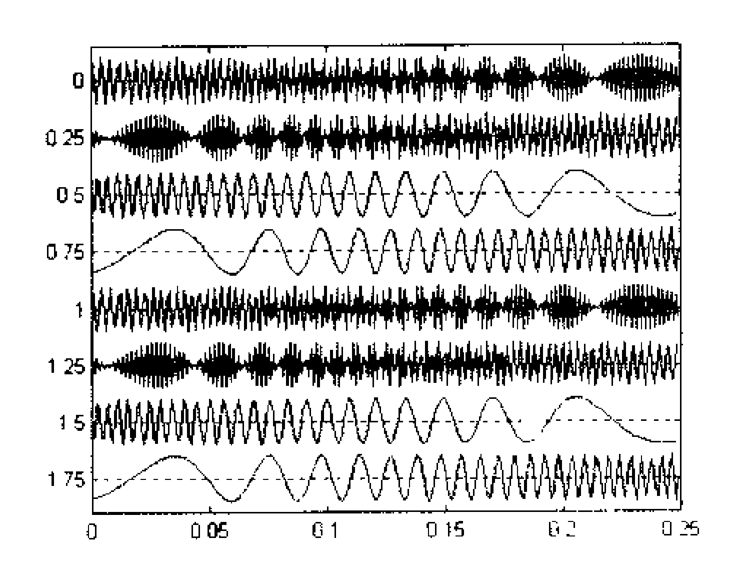


图 2.47 频率调制信号的带状图

9. xcorr2

功能：二维互相关系数。

格式：

```
C=xcorr2(A)
C=xcorr2(A, B)
```

说明：

C=xcorr2(A，B)可得到矩阵 A 和 B 的互相关系数，它是一维互相关系数 xcorr 的推广。

xcorr2(A)可得到矩阵 A 的自相关系数，它等效于 xcorr2(A，A)。

参见：conv2，filter2，xcorr

第3章

信号处理系统分析与设计

在第 1 章信号处理基本理论和第 2 章信号处理工具箱函数的基础上，我们可利用 MATLAB 及其工具箱提供的 M 函数来解决信号处理领域的问题。

本章的每一节与第 1 章的节一一对应。在每一节中，首先给出一些扩展函数，这些函数可使设计得以简化，其理论源于第 1 章。然后给出许多设计示例，每个例于都给出 MATLAB 程序，它可在 MATLAB V4.21c 中直接执行，执行后产生的结果绝大部分以图形表示。

3.1　离散信号与系统

MATLAB 的扩展函数有 8 种。

1. 单位取样序列

```
impseq.m
function [x, n]=impseq(n0, n1, n2)
% Generates x(n)=delta(n-n0); n1<=n, n0<=n2
% -----------------
% [x, n]=impseq(n0, n1, n2)
%
if ((no<n1)|(n2>n2)|(n1>n2)
        error('arguments must satisfy n1<=n0<=n2')
end
n=[n1:n2];
x=[(n-n0)==0];
```

2. 单位阶跃序列

```
stepseq. m
function [x, n]=stepseq(n0, n1, n2)
% Generates x(n)=u(n-n0); n1<=n, n0<=n2
% -----------------
% [x, n]=stepseq(n0, n1, n2)
%
if ((no<n1)|(n0>n2)|(n1>n2))
        error('arguments must satisfy n1<=n0<=n2')
end
n=[n1:n2];
x=[(n-n0)>=0];
```

3. 信号加

```
sigadd. m
function [y, n]=sigadd(x1, n1, x2, n2)
% implements y(n)=x1(n)+x2(n)
% -----------------
% [y, n]=sigadd(x1, n1, x2, n2)
% y=sum sequence over n, which includes n1 and n2
% x1=first sequence over n1
% x2=second sequence over n2(n2 can be different from n1)
%
n=min(min(n1), min(n2):max(max(n1), max(n2));
y1=zeros(1, length(n)); y2=y1;
y1(find((n>=min(n1)&(n<=max(n1))==1))=x1;
y2(find((n>=min(n2))&(n<=max(n2))==1))=x2;
y=y1+y2;
```

4. 信号乘

```
sigmult. m
function [y, n]=sigmult(x1, n1, x2, n2)
% implements y(n)=x1(n) * x2(n)
% -----------------
% [y, n]=sigmult(x1, n1, x2, n2)
% y=product sequence over n, which includes n1 and n2
% x1=first sequence over n1
```

```
% x2=second sequenc eover n2(n2 can be different from n1)
%
n=min(min(n1), min(n2):max(max(n1), max(n2));
y1=zeros(1, length(n)); y2=y1;
y1(find((n>=min(n1)&(n<=max(n1))==1))=x1;
y2(find((n>=min(n2))&(n<=max(n2))==1))=x2;
y=y1. * y2;
```

5. 移位

```
sigshift. m
function [y, n]=sigshift(x, m, n0)
% implements y(n)=x(n-n0)
% -----------------
% [y, n]=sigshift(x, m, n0)
%
n=m+n0; y=x;
```

6. 折叠

```
sigfold. m
function [y, n]=sigfold(x, n)
% implements y(n)=x(-n)
% -----------------
% [y, n]=sigsfold(x, n)
%
y=fliplr(x); n=-fliplr(n);
```

7. 奇偶综合

```
evenodd. m
function [xe, xo, m]=evenodd(x, n)
% Real signal decomposition into even and odd parts
% -----------------
% [xe, xo, m]=evenodd(x, n)
%
if any (imag(x)=0)
        error('x is not a real sequence')
end
m= -fliplr(n);
m1=min([m, n]); m2=max([m, n]); m=m1:m2;
```

```
nm=n(1)-m(1); n1=1:length(n);
x1=zeros(1, length(m));
x1(n1+nm)=x; x=x1;
xe=0.5*(x+fliplr(x));
xo=0.5*(x-fliplr(x));
```

8. 修改后的卷积

```
conv_m.m
function [y, ny]=conv_m(x, nx, h, nh)
% Modified convolution routine for signal processing
% ----------------
% [y, ny]=conv m(x, nx, h, nh)
% y=convolution result
% ny=support of y
% x=first signal on support nx
% nx=support of x
% h=second signal on support nh
% nh=support of h
%
nyb=nx(1)+nh(1); nye=nx(lengthx))+nh(lengthh));
ny=[nyb:nye];
y=conv(x, h);
```

例 3.1 产生下列序列并绘出离散图：

a. $x(n)=2\delta(n+2)-\delta(n-4) \qquad -5\leqslant n\leqslant 5$

b. $x(n)=n[u(n)-u(n-10)]+10e^{-0.3(n-10)}[u(n-10)-u(n-20)] \quad 0\leqslant n\leqslant 20$

c. $x(n)=\cos(0.04\pi n)+0.2w(n) \qquad 0\leqslant n\leqslant 50$

其中 $w(n)$是均值为零、方差为 1 的白噪声序列

d. $\tilde{x}(n)=[x1\ x1\ x1\ x1]$，其中 $x1=[5\ 4\ 3\ 2\ 1]$

解 利用 MATLAB 及信号处理工具箱函数，再加上前面构造的几个函数如 sigadd，sigmult 等，可很容易编写出可直接执行的 MATLAB 程序 ex3001.m：

```
%
% Example 3.1
%
%
% a) x(n)=2*delta(n+2) - delta(n-4), -5<=n<=5
%
figure(1); clf
n=[-5:5];
```

```
x=2*impseq(-2, -5, 5)-impseq(4, -5, 5);
subplot(2, 2, 1); stem(n, x); title('Sequence in Example 3.1a')
xlabel('n'); ylabel('x(n)'); axis([-5, 5, -2, 3])
%
% b) x(n)=n[u(n)-u(n-10)]+10*exp(-0.3(n-10))(u(n-10)
% -u(n-20)); 0<=n<=20
%
n=[0:20];
x1=n.*(stepseq(0, 0, 20)-stepseq(10, 0, 20));
x2=10*exp(-0.3*(n-10)).*(stepseq(10, 0, 20)-stepseq(20, 0, 20));
x=x1+x2;
subplot(2, 2, 2); stem(n, x);
title('Sequence in Example 3.1b')
xlabel('n'); ylabel('x(n)'); axis([0, 20, -1, 11])
%
% c) x(n)=cos(0.04*pi*n) + 0.2*w(n); 0<=n<=50
%
n=[0:50];
x=cos(0.04*pi*n)+0.2*randn(size(n));
subplot(2, 2, 3); stem(n, x); title('Sequence in Example 3.1c')
xlabel('n'); ylabel('x(n)'); axis([0, 50, -1.4, 1.4])
%
% d) x(n)={..., 5, 4, 3, 2, 1, 5, 4, 3, 2, 1, ...}; -10<=n<=9
%
n=[-10:9];
x=[5, 4, 3, 2, 1];
xtilde=x' * ones(1, 4);
xtilde=(xtilde(:))';
subplot(2, 2, 4); stem(n, xtilde); title('Sequence in Example 3.1d')
xlabel('n'); ylabel('xtilde(n)'); axis([-10, 9, -1, 6])
```

在 MATLAB 下执行

```
ex3001
```

可得到如图 3.1 所示的结果。

例 3.2 产生复信号

$$x(n) = e^{(-0.1+j0.3)n} \qquad -10 \leqslant n \leqslant 10$$

并画出复序列 x(n)的实部、虚部、幅值和相位离散图。

解 MATLAB 程序为 ex3002.m:

```
% Example 3.2
%
```

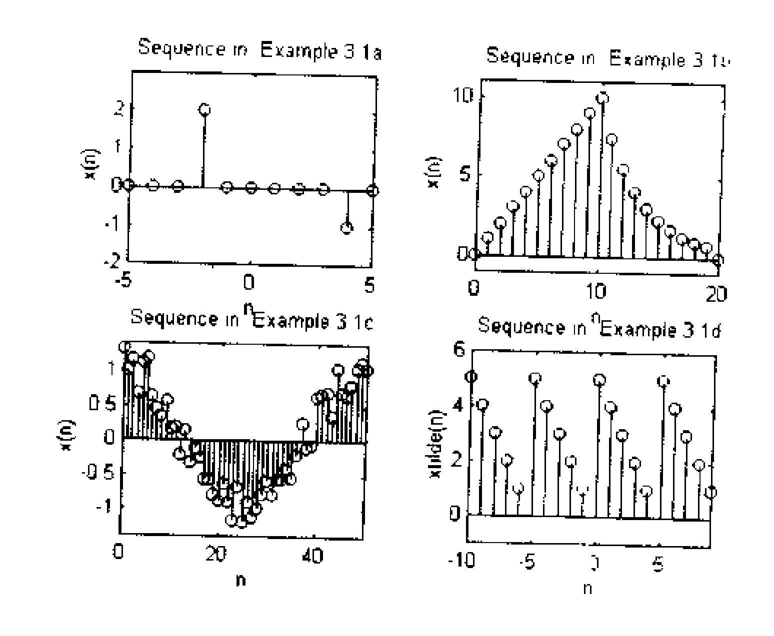
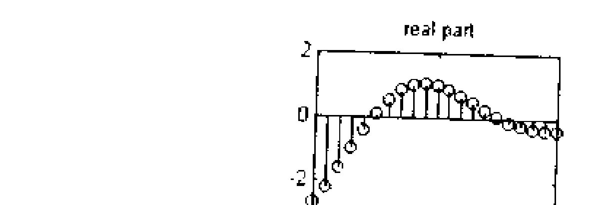
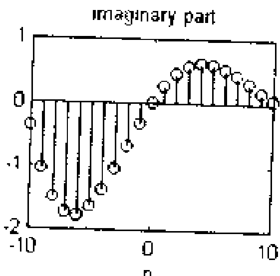


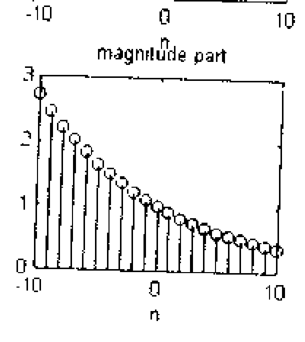
图 3.1 离散信号图

```
%    x(n)=exp((—0.1+j0.3)n), —10 <= n <= 10;
%
figure(1); clf
n=[—10:1:10]; alpha=—0.1+0.3j;
x=exp(alpha * n);
subplot(2, 2, 1); stem(n, real(x)); title('real part'); xlabel('n')
subplot(2, 2, 2); stem(n, imag(x)); title('imaginary part'); xlabel('n')
subplot(2, 2, 3); stem(n, abs(x)); title('magnitude part'); xlabel('n')
subplot(2, 2, 4); stem(n, (180/pi) * angle(x)); title('phase part'); xlabel('n')
```

执行结果如图 3.2 所示。







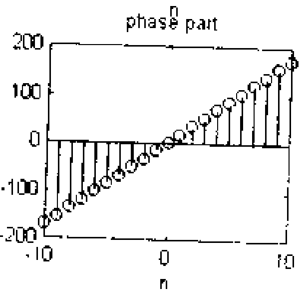


图 3.2 复信号图

例 3.3 已知 $x(n)=u(n)-u(n-10)$，要求将它分解成奇偶序列。

解 MATLAB 程序为 ex3003.m：

```
% Example 3.3
%
% x(n)=u(n)-u(n-10);
%
n=[0:10];
x=stepseq(0, 0, 10)-stepseq(10, 0, 10);
[xe, xo, m]=evenodd(x, n);
figure(1)
subplot(2, 2, 1); stem(n, x); title('Rectangular pulse')
xlabel('n'); ylabel('x(n)'); axis([-10, 10, -1.2, 1.2])
subplot(2, 2, 2); stem(m, xe); title('Even Part')
xlabel('n'); ylabel('xe(n)'); axis([-10, 10, -1.2, 1.2])
subplot(2, 2, 4); stem(m, xo); title('Odd Part')
xlabel('n'); ylabel('xe(n)'); axis([-10, 10, -1.2, 1.2])
```

执行结果如图 3.3 所示。

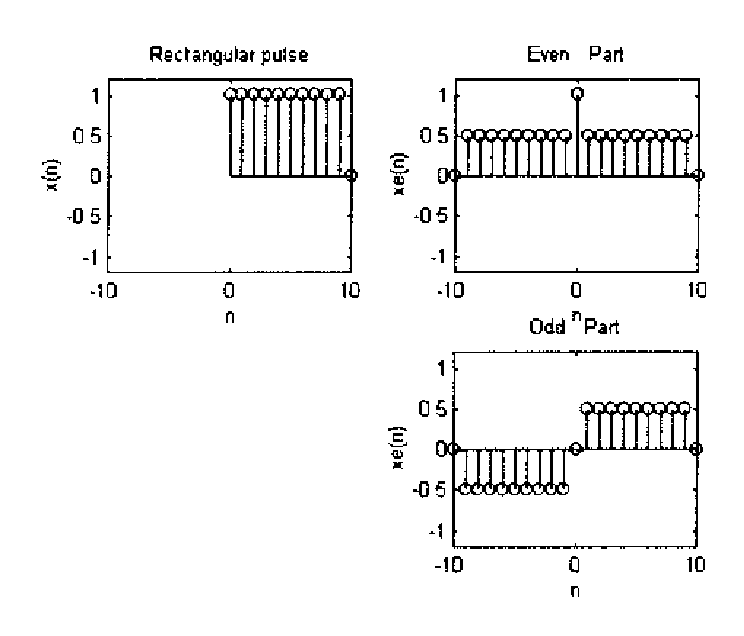


图 3.3 序列的奇偶分解

例 3.4 设线性时不变(LTI)系统的冲激响应为

$$h(n) = (0.9)^n u(n)$$

输入序列为

$$x(n) = u(n) - u(n-10)$$

求系统输出 $y(n)$。

解 系统输出 $y(n)$为输入 $x(n)$与系统冲激响应 $h(n)$的卷积，可直接采用 conv.m 函

数求出输出序列。MATLAB 程序为 ex3004.m：

```
% Example 3.4
%
% x(n)=[u(n)-u(n-10]; h(n)=(0.9)^n*u(n) n=0:40
% y(n)=conv(x, h)
%
n=-5:50;
u1=stepseq(0, -5, 50); u2=stepseq(10, -5, 50);

% input x(n)
x=u1-u2;

% impulse response h(n)
h=((0.9).^n).*u1;
figure(1)
subplot(3, 1, 1); stem(n, x); axis([-5, 50, 0, 2])
title('Input Sequence')
ylabel('x(n)')
subplot(3, 1, 2); stem(n, h); axis([-5, 50, 0, 2])
gtext('Impulse Response')
ylabel('h(n)');

% output response
[y, ny]=conv_m(x, n, h, n);
subplot(3, 1, 3); stem(ny, y); axis([-5, 50, 0, 8])
gtext('Output Sequence')
xlabel('n'), ylabel('y(n)')
```

执行结果给出这 3 个序列的离散图，如图 3.4 所示。

例 3.5 设两序列分别为

$$x = [3\ 11\ 7\ 0\ -1\ 4\ 2]$$
$$n = [-3\ -2\ -1\ 0\ 1\ 2\ 3\ ]$$
$$y(k) = x(k-2) + w(k) \quad k \in n$$

其中 w(k)是均值为 0、方差为 1 的高斯白噪声序列。计算序列 x(n)和 y(n)之间的相关系数。

解 据定义，相关系数为

$$r_{yx}(l) = y(l) * x(l)$$

这样就可以利用 conv_m 函数进行计算。

由于 y 序列是 x 序列经移位后加上一白噪声序列，因此 xy 之间在 l=2 处具有强烈的相关性，为说明问题，我们给出了两种白噪声序列下的相关性。

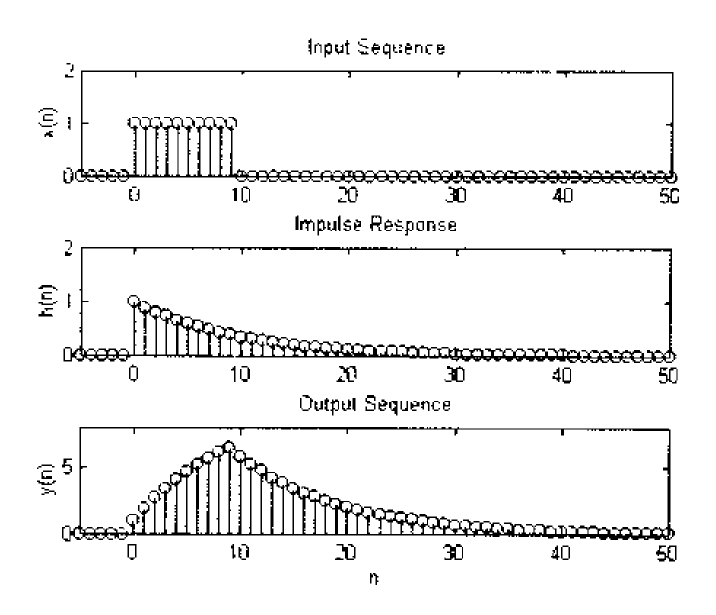


图 3.4 LIT 系统的输出

MATLAB 程序为 ex3005.m：

```
% Example 3.5
%
% x(n)=[3, 11, 7, 0, -1, 4, 2]; nx=[-3:3]
% y(n)=x(n-2)+w(n)
% ryx=cross(y, x)
%
% === noise sequence 1 ===
%
x=[3, 11, 7, 0, -1, 4, 2]; nx=[-3:3];        % given signal x(n)
[y, ny]=sigshift(x, nx, 2);
w=randn(1, length(y)); nw=ny;                % generate w(n)
[y, ny]=sigadd(y, ny, w, nw);
[x, nx]=sigfold(x, nx);
[rxy, nrxy]=conv_m(y, ny, x, nx);
subplot(1, 1, 1)
subplot(2, 1, 1); stem(nrxy, rxy)
axis([-4, 8, -50, 250]); xlabel('lag variable l')
ylabel('rxy'); title('Crosscorrelation: noise sequence 1')
gtext('Maximum')

%
% === noise sequence 2 ===
x=[3, 11, 7, 0, -1, 4, 2]; nx=[-3:3];        % given signal x(n)
```

```
[y, ny]=sigshift(x, nx, 2);
w=randn(1, length(y)); nw=ny;                    % generate w(n)
[y, ny]=sigadd(y, ny, w, nw);
[x, nx]=sigfold(x, nx);
[rxy, nrxy]=conv_m(y, ny, x, nx);
subplot(2, 1, 2); stem(nrxy, rxy)
gtext('Maximum')
axis([-4, 8, -50, 250]); xlabel('lag variable l')
ylabel('rxy'); title('Crosscorrelation: noise sequence 2')
```

这可得到如图 3.5 所示的相关序列。

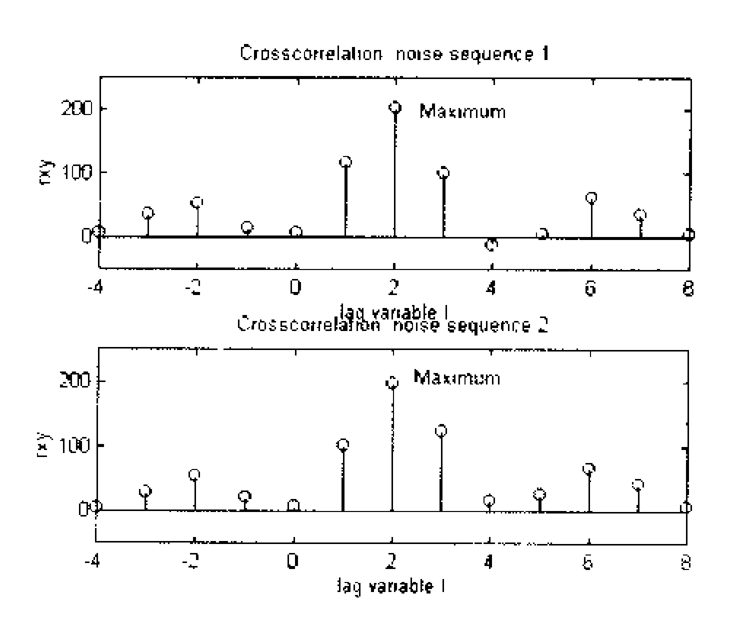


图 3.5 不同噪声下的相关序列

例 3.6 设离散系统可由下列差分方程表示

$$y(n) - y(n-1) + 0.9y(n-2) = x(n) \quad \forall n$$

(1) 计算 n=-20, …, 100 上的冲激响应;

(2) 计算 n=-20, …, 100 上的单位阶跃响应。

解 这可以利用 filter 函数直接设计。MATLAB 程序为 ex3006.m:

```
% Example 3.6
%
a=[1, -1, 0.9]; b=1;
%
% Part (2)
%
x=impseq(0, -20, 120); n=[-20:120];
h=filter(b, a, x);
subplot(2, 1, 1); stem(n, h)
```

```
axis([-20, 120, -1.1, 1.1])
title('Impulse Response'); xlabel('n'); ylabel('h(n)')
%
% Part (1)
%
x=stepseq(0, -20, 120);
s=filter(b, a, x);
subplot(2, 1, 2); stem(n, s)
axis([-20, 120, -.5, 2.5])
title('Step Response'); xlabel('n'); ylabel('s(n)')
```

系统冲激响应和阶跃响应如图 3.6 所示。

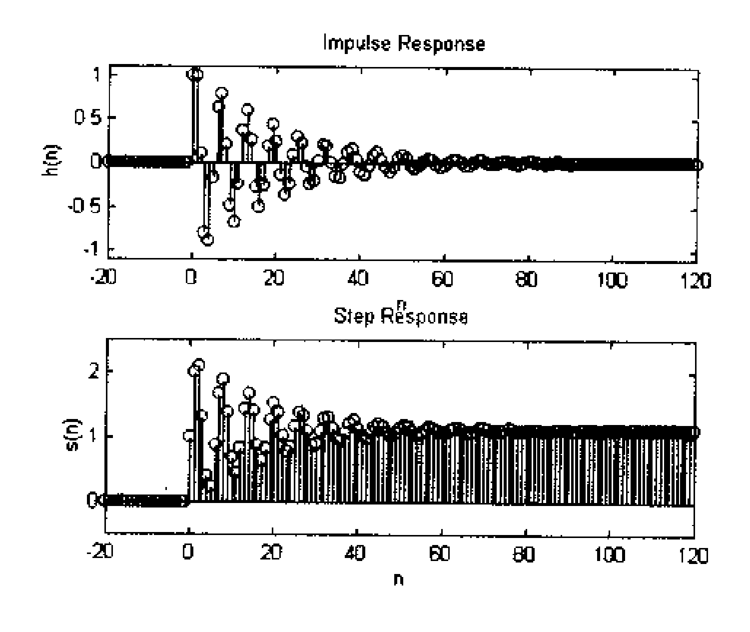


图 3.6 系统的冲激响应和阶跃响应

3.2 离散时间傅里叶分析

3.2.1 离散时间傅里叶变换(DFT)

例 3.7 已知 $x(n)=(0.9e^{j\frac{\pi}{3}})^n$，$0\leqslant n\leqslant 10$，求这一有限时宽序列 x(n)的傅里叶变换 $X(e^{j\omega})$。

解 根据定义有

$$X(e^{j\omega}) = \sum_{n=0}^{10} e^{-j\omega n} x(n)$$

记矢量 $x=[x(0)\ x(1)\ \cdots\ x(n)]$，取 $\omega=\frac{\pi}{M}k$，$k=0, 1, \cdots, M$，即将$[0, \pi]$均分成 M+1

点，这时记矩阵 $W \triangleq \{e^{-j\frac{\pi}{M}kn}, 0 \leqslant n \leqslant 10, k=0, 1, \cdots, M\}$，矢量 $X=\{X(e^{j\omega})\}$，则上式可写成

$$X = Wx$$

如果记矢量 $n=[0, 1, \cdots, 10]$，$k=[0, 1, \cdots, M]$，则有

$$W = \left[\exp\left(-j\frac{\pi}{M}k^T n\right)\right]$$

因此得到求 x(n)的傅里叶变换的 MATLAB 程序为 ex3007.m：

```
% Example 3.7
%
n=0:10; x=(0.9*exp(j*pi/3)).^n;
k=-200:200; w=(pi/100)*k;
X=x * (exp(-j*pi/100)) .^ (n'*k);
magX=abs(X); angX =angle(X);
figure(1)
subplot(2, 1, 1); plot(w/pi, magX, 'w'); grid
axis([-2, 2, 0, 8])
xlabel('frequency in units of pi'); ylabel('|X|')
title('Magnitude Part ')
subplot(2, 1, 2); plot(w/pi, angX/pi, 'w'); grid
axis([-2, 2, -1, 1])
xlabel('frequency in units of pi'); ylabel('radians/pi')
title('Angle Part ')
```

这可得到如图 3.7 所示的幅频相频曲线。由图可以看出，$X(e^{j\omega})$是周期为 2π 的周期信号，但不是复共轭对称，实际上 x(n)为复序列，这与理论结果相一致。

例 3.8 设 $x(n)=2^n$，$-10 \leqslant n \leqslant 10$，求相应的 $X(e^{j\omega})$。

解 方法与例 3.7 相同，MATLAB 程序为 ex3008.m：

```
% Example 3.8
%
n=-5:5; x=(-0.9).^n;
k=-200:200; w=(pi/100)*k;
X=x * (exp(-j*pi/100)) .^ (n'*k);
magX=abs(X); angX =angle(X);
subplot(2, 1, 1); plot(w/pi, magX, 'w'); grid
axis([-2, 2, 0, 15])
xlabel('frequency in units of pi'); ylabel('|X|')
gtext('Magnitude Part')
subplot(2, 1, 2); plot(w/pi, angX, 'w')/pi; grid
axis([-2, 2, -4, 4])
xlabel('frequency in units of pi'); ylabel('radians/pi')
```

gtext('Angle Part')

这得到了如图 3.8 所示的幅频相频特性。由图可以看出，$X(e^{j\omega})$不仅是周期信号，而且是复共轭对称信号，这是因为 x(n)为实序列，这与理论结果一致。

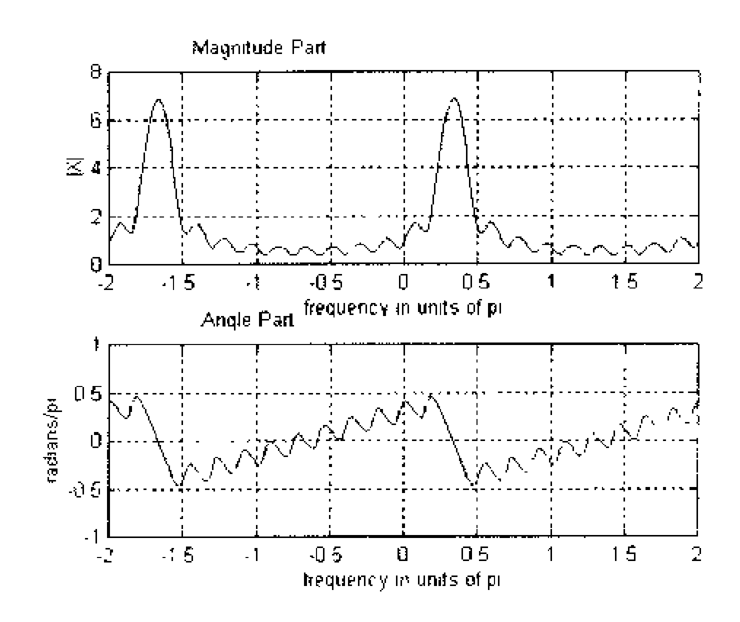


图 3.7 幅频和相频曲线

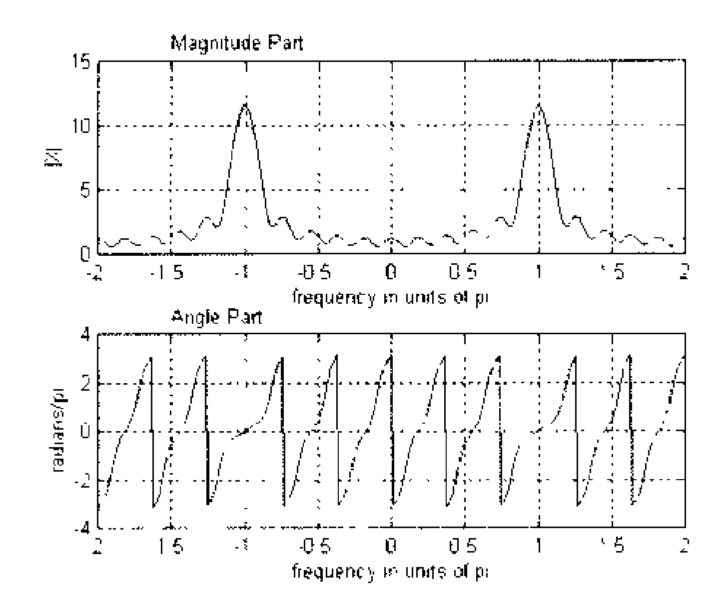


图 3.8 幅频相频曲线

3.2.2 DFT 特性

例 3.9 设有两序列

$$x(n) = \cos\left(\frac{\pi n}{2}\right) \quad 0 \leqslant n \leqslant 100$$

$$y(n) = e^{j\pi n/4}x(n) \quad 0 \leqslant n \leqslant 100$$

试比较 $X(e^{j\omega})$和 $Y(e^{j\omega})$。

解 MATLAB 程序为 ex3009.m：

```
% Example 3.9
%
n=0:100; x=cos(pi * n/2);
k=-100:100; w=(pi/100) * k;          % frequency between -pi and +pi
X=x * (exp(-j * pi/100)).^(n' * k);  % DFT of x
%
y=exp(j * pi * n/4). * x;
Y=y * (exp(-j * pi/100)).^(n' * k);  % DFT of y
% Graphical verification
figure(1)
subplot(2, 2, 1); plot(w/pi, abs(X), 'w'); grid; axis([-1, 1, 0, 60])
xlabel('frequency in pi units'); ylabel('|X|')
gtext('Magnitude of X')
subplot(2, 2, 2); plot(w/pi, angle(X)/pi, 'w'); grid; axis([-1, 1, -1, 1])
xlabel('frequency in pi units'); ylabel('radiands/pi')
gtext('Angle of X')
subplot(2, 2, 3); plot(w/pi, abs(Y), 'w'); grid; axis([-1, 1, 0, 60])
xlabel('frequency in pi units'); ylabel('|Y|')
gtext('Magnitude of Y')
subplot(2, 2, 4); plot(w/pi, angle(Y)/pi, 'w'); grid; axis([-1, 1, -1, 1])
xlabel('frequency in pi units'); ylabel('radians/pi')
gtext('Angle of Y')
```

这可得到如图 3.9 所示的曲线。由图可以看出，$Y(e^{j\omega})$是$X(e^{j\omega})$移位 $\pi/4$ 后的结果。

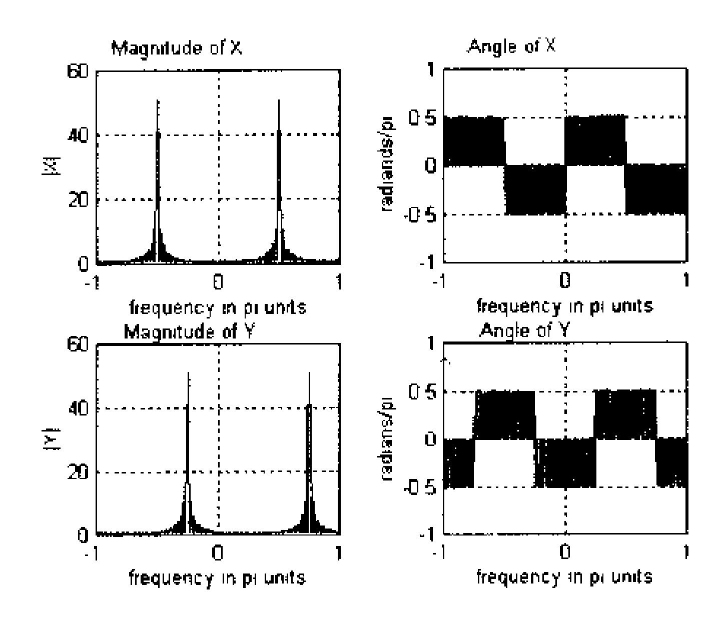


图 3.9 移频特性

例 3.10 实序列

$$x(n) = \sin\left(\frac{\pi n}{2}\right) \quad -5 \leqslant n \leqslant 10$$

求出 $X(e^{j\omega})$的实部和虚部，同时求出奇偶部分相应的 $X_e(e^{j\omega})$和 $X_o(e^{j\omega})$。

解 MATLAB 程序为 ex3010.m：

```
% Example 3.10
%
n=-5:10; x=sin(pi*n/2);
k=-100:100; w=(pi/100)*k;
      % frequency between -pi and +pi
X=x * (exp(-j*pi/100)).^(n'*k);          % DFT of x
% signal decomposition
[xe, xo, m]=evenodd(x, n);               % even and odd parts
XE=xe * (exp(-j*pi/100)).^(m'*k);        % DFT of xe
XO=xo * (exp(-j*pi/100)).^(m'*k);        % DFT of xo
XR=real(X);
XI=imag(X);
% graphical verification
figure(1)
subplot(2, 2, 1); plot(w/pi, XR, 'w'); grid; axis([-1, 1, -2, 2])
xlabel('frequency in pi units'); ylabel('Re(X)');
gtext('Real part of X')
subplot(2, 2, 2); plot(w/pi, XI, 'w'); grid; axis([-1, 1, -10, 10])
xlabel('frequency in pi units'); ylabel('Im(X)');
gtext('Imaginary part of X')
subplot(2, 2, 3); plot(w/pi, real(XE), 'w'); grid; axis([-1, 1, -2, 2])
xlabel('frequency in pi units'); ylabel('XE');
gtext('Transform of even part')
subplot(2, 2, 4); plot(w/pi, imag(XO), 'w'); grid; axis([-1, 1, -10, 10])
xlabel('frequency in pi units'); ylabel('XO');
gtext('Transform of odd part')
```

这可得到如图 3.10 所示的曲线。由图可以看出，$\mathscr{F}([x_e(n)]=\mathrm{Re}[X(e^{j\omega})]$，$\mathscr{F}[x_o(n)]=\mathrm{Im}[X(e^{j\omega})]$。

3.2.3 LTI 系统频域表示

例 3.11 差分方程

$$\begin{aligned} y(n) = &\ 0.0181x(n) + 0.0543x(n-1) + 0.0543x(n-2) \\ &+ 0.0181x(n-3) + 1.76y(n-1) - 1.1829y(n-2) \\ &+ 0.2781y(n-3) \end{aligned}$$

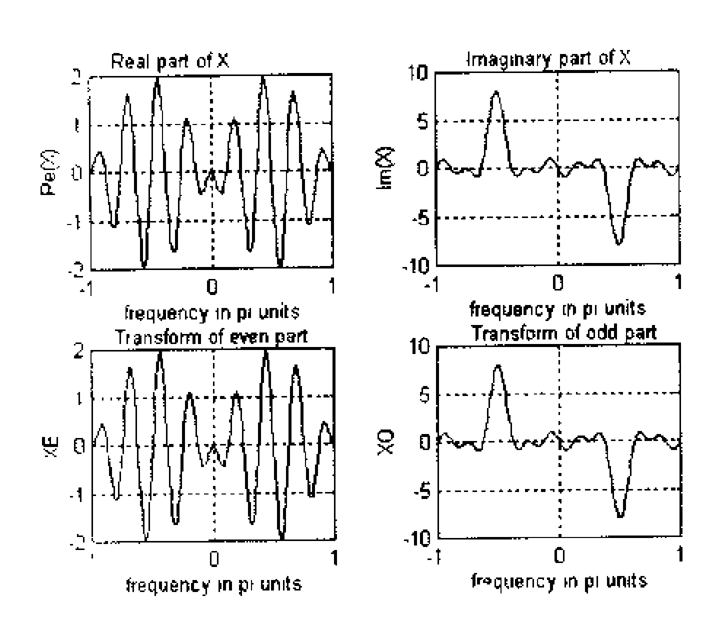


图 3.10 实序列的特性曲线

表示一个 3 阶的低通滤波器，试绘出滤波器的幅频和相频特性。

解 MATLAB 程序为 ex3011.m：

```
% Example 3.11
%
b=[0.0181,  0.0543, 0.0543,  0.0181];
a=[1.0000, -1.7600, 1.1829, -0.2781];
m=0:length(b)-1; l=0:length(a)-1;
K=500; k=1:1:K;
w=pi*k/K;                          % [0, pi] axis divided into 501 points.
num=b * exp(-j*m'*w);              % Numerator calculations
den=a * exp(-j*l'*w);              % Denominator calculations
H=num ./ den;
magH=abs(H); angH=angle(H);
figure(1);
subplot(2, 1, 1); plot(w/pi, magH, 'w'); grid; axis([0, 1, 0, 1])
xlabel('frequency in pi units'); ylabel('|H|');
gtext('Magnitude Response');
subplot(2, 1, 2); plot(w/pi, angH/pi, 'w'); grid
xlabel('frequency in pi units'); ylabel('Phase in pi Radians');
gtext('Phase Response');
```

这产生了如图 3.11 所示的低通滤波器幅频相频特性曲线。

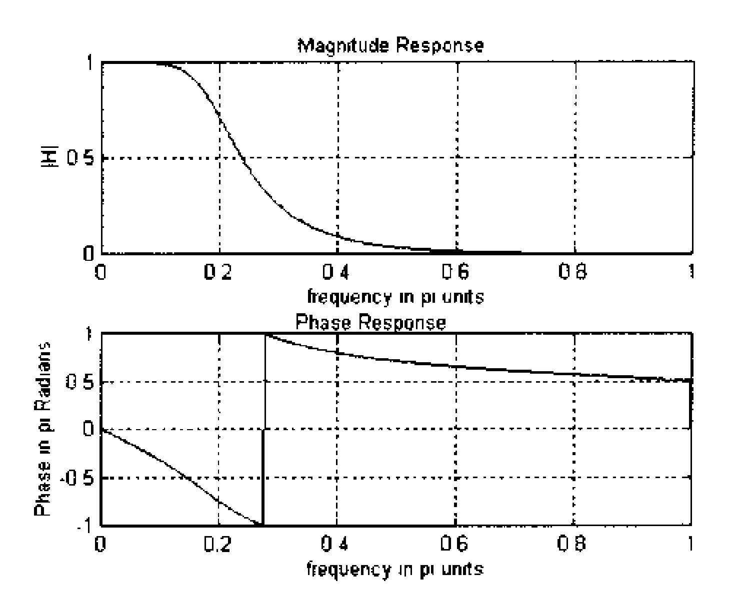


图 3.11　低通滤波器幅频相频特性曲线

3.2.4　模拟信号的取样和重构

例 3.12　设 $x_a(t)=e^{-1000|t|}$，求其傅里叶变换 $X_a(j\omega)$。

解　
$$X_a(j\omega)=\int_{-\infty}^{\infty}x_a(t)e^{-j\omega t}dt$$
$$=\int_{-\infty}^{0}e^{1000t}e^{-j\omega t}dt+\int_{0}^{\infty}e^{1000t}e^{-j\omega t}dt$$
$$=\frac{0.002}{1+\left(\dfrac{\omega}{1000}\right)^2}$$

严格说来，在 MATLAB 中不使用 symbolic 工具箱是不能分析模拟信号的，但当以充分小的时间间隔取样 $x_a(t)$ 时，可产生平滑的图形，当包含足够长的时间时，可显示出所有的模式，这样就可以近似地进行分析。

在这个例子中，由上式知在 10^{-5} 精度下，$x_a(t)$ 为 $F_0=2000$ 的带限信号，因此若取

$$\Delta t=5\times10^{-5}\ll\frac{1}{2\times2000}=25\times10^{-5}$$

这样就可得到 $x_a(t)$ 的逼近序列 $x_G(m)$。

MATLAB 程序为 ex3012.m：

```
% Example 3.12
%
% Analog Signal
Dt=0.00005; t=-0.005:Dt:0.005;
xa=exp(-1000*abs(t));
%
% Continuous-time Fourier Transform
Wmax=2*pi*2000;
```

```
K=500; k=0:1:K; W=k * Wmax/K;
Xa=xa * exp(-j * t' * W) * Dt;
Xa=real(Xa);
W=[-fliplr(W), W(2:501)];
Xa=[fliplr(Xa), Xa(2:501)];
figure(1)
subplot(2, 1, 1); plot(t * 1000, xa, 'w');
xlabel('t in msec.'); ylabel('xa(t)')
gtext('Analog Signal')
subplot(2, 1, 2); plot(W/(2 * pi * 1000), Xa * 1000, 'w');
xlabel('Frequency in KHz'); ylabel('Xa(jW) * 1000')
gtext('Continuous-time Fourier Transform')
```

这就得到了如图 3.12 所示的曲线。

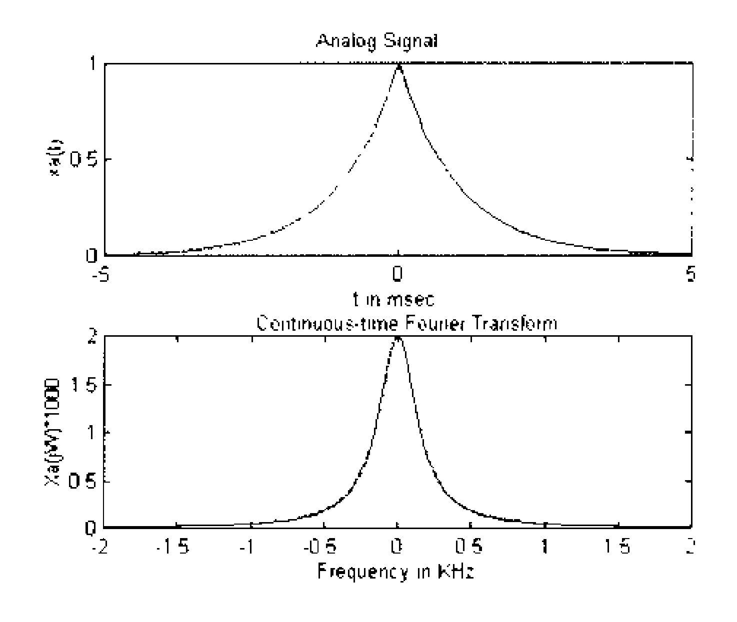


图 3.12 $x_a(t)$和$X_a(j\omega)$曲线

例 3.13 以例 3.12 中的 $x_a(t)$说明取样速率对频域特性的影响：

(1) 取样频率 Fs=5000 Hz，绘出 $X_1(e^{j\omega})$曲线；

(2) 取样频率 Fs=1000 Hz，绘出 $X_2(e^{j\omega})$曲线。

解 由于 $x_a(t)$的带宽为 2000 Hz，因此 Nyquist 速率为 4000 Hz，因此情况(1)不会发生混淆，而情况(2)会产生混叠现象。

MATLAB 程序为 ex3013.m：

```
% Example 3.13
%
% Analog Signal
Dt=0.00005; t=-0.005:Dt:0.005;
xa=exp(-1000 * abs(t));
```

```
%
% Discrete-time Signal
Ts=0.0002; n=-25:1:25;
x=exp(-1000 * abs(n * Ts));
%
% Discrete-time Fourier transform
K=500; k=0:1:K;
w=pi * k/K;
X=x * exp(-j * n' * w);
X=real(X);
w=[-fliplr(w), w(2:K+1)];
X=[fliplr(X), X(2:K+1)];
figure(1)
subplot(2, 1, 1); plot(t * 1000, xa, 'w');
xlabel('t in msec.'); ylabel('x1(t)')
gtext('Discrete Signal'); hold on
stem(n * Ts * 1000, x); hold off
subplot(2, 1, 2); plot(w/pi, X, 'w');
xlabel('Frequency in pi units'); ylabel('X1(w)')
gtext('Discrete-time Fourier Transform')
gtext('Ts= 0.2 msec')
```

这可得到如图 3.13 所示的曲线。与图 3.12 比较，可见 $X_1(e^{j\omega})$与 $X_a(j\omega)$是相同的。对 MATLAB 程序文件稍加修改，可产生如图 3.14 所示的 $X_2(e^{j\omega})$图形，从图中可以看出 $X_2(e^{j\omega})$与 $X_a(j\omega)$不同，即产生了混叠现象。

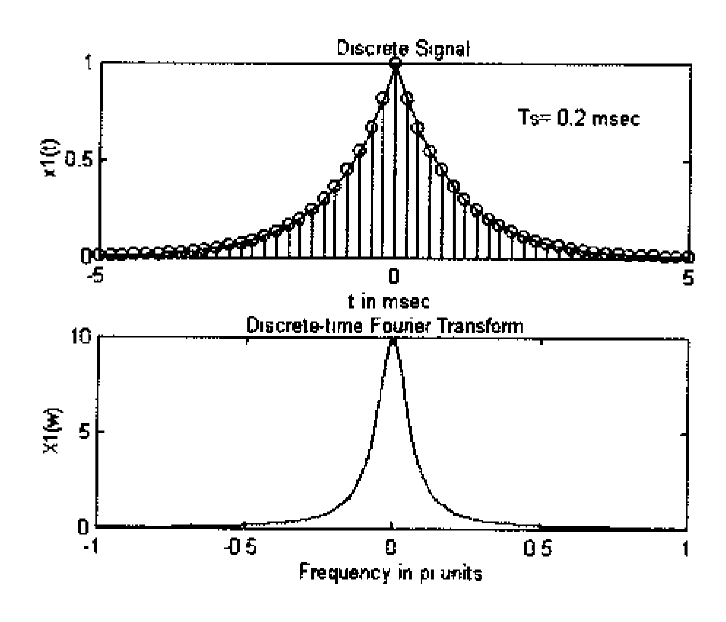


图 3.13　F_s=5 000 Hz 时的 $x_1(n)$与 $X_1(e^{j\omega})$图形

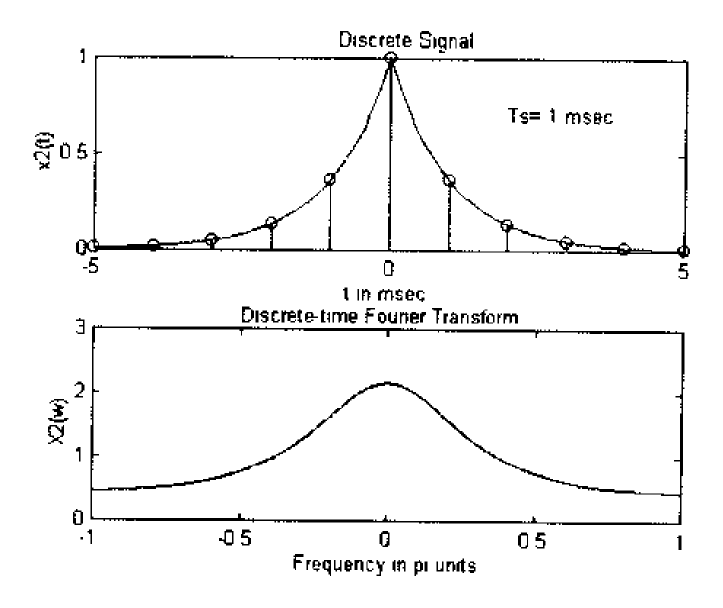


图 3.14 F_s=1000 Hz 时的 $x_2(n)$与 $X_2(e^{j\omega})$图形

例 3.14 对例 3.13 中产生的 $x_1(n)$序列，采用 sinc 和 3 次样条内插重构 $x_a(t)$。

解 MATLAB 程序为 ex3014.m：

```
% Example 3.14
%
% Discrete-time Signal x1(n)
Ts=0.0002; Fs=1/Ts; n=-25:1:25; nTs=n * Ts;
x=exp(-1000 * abs(nTs));

% Analog Signal reconstruction(using sinc function)
Dt=0.00005;
t=-0.005:Dt:0.005;
xa=x * sinc(Fs * (ones(length(nTs), 1) * t-nTs' * ones(1, length(t))));
error=max(abs(xa - exp(-1000 * abs(t))))
figure(1)
subplot(2, 1, 1); plot(t * 1000, xa, 'w');
gtext('t in msec'); ylabel('xa(t)')
title('Reconstructed Signal from x1(n) using sinc function'); hold on
stem(n * Ts * 1000, x); hold off

% Analog Signal reconstruction(using cubic spline function)
xa=spline(nTs, x, t);
error=max(abs(xa - exp(-1000 * abs(t))))
subplot(2, 1, 2); plot(t * 1000, xa, 'w');
gtext('t in msec'); ylabel('xa(t)')
```

```
title('Reconstructed Signal from x1(n) using cubic spline function'); hold on
stem(n * Ts * 1000, x); hold off
```

执行后得采用 sinc 函数时重构信号与原信号的最大误差为 0.0363，采用 3 次样条函数时的最大误差为 0.0317，这说明重构的精度已相当不错。重构信号如图 3.15 所示，这可与图 3.13 进行对比。

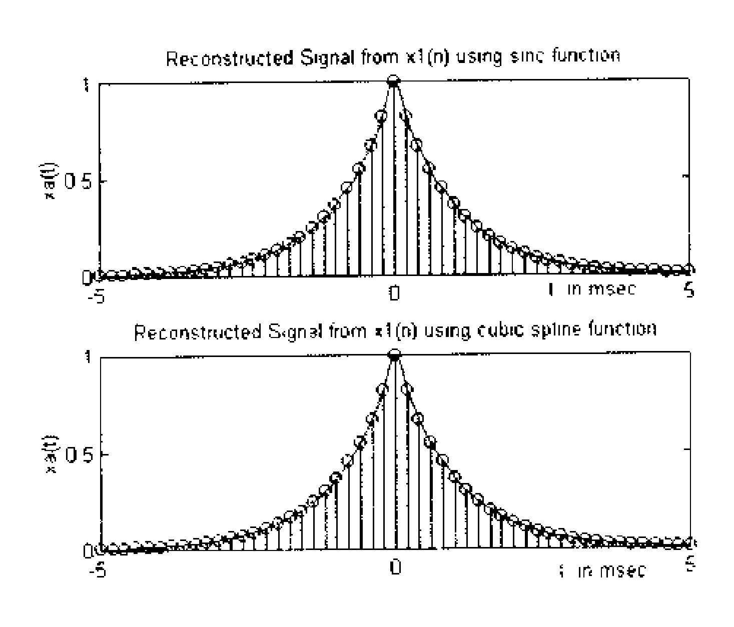


图 3.15　信号重构

例 3.15　对例 3.13 中产生的 $x_2(n)$序列，采用 sinc 和 3 次样条内插重构 $x_a(t)$。

解　MATLAB 程序为 ex3015.m：

```
% Example 3.15
%
% Discrete-time Signal x1(n)
Ts=0.001; Fs=1/Ts; n=-5:1:5; nTs=n * Ts;
x=exp(-1000 * abs(nTs));

% Analog Signal reconstruction(using sinc function)
Dt=0.00005;
t=-0.005:Dt:0.005;
xa=x * sinc(Fs * (ones(length(nTs), 1) * t-nTs' * ones(1, length(t))));
error=max(abs(xa - exp(-1000 * abs(t))))
figure(1)
subplot(2, 1, 1); plot(t * 1000, xa, 'w');
gtext('t in msec'); ylabel('xa(t)')
title('Reconstructed Signal from x1(n) using sinc function'); hold on
stem(n * Ts * 1000, x); hold off
```

```
% Analog Signal reconstruction(using cubic spline function)
xa=spline(nTs, x, t);
error=max(abs(xa - exp(-1000 * abs(t))))
subplot(2, 1, 2); plot(t * 1000, xa, 'w');
gtext('t in msec'); ylabel('xa(t)')
title('Reconstructed Signal from x1(n) using cubic spline function'); hold on
stem(n * Ts * 1000, x); hold off
```

执行后得到采用 sinc 函数时重构信号与原信号的最大误差为 0.1852，采用 3 次样条函数时的最大误差为 0.1679，这说明重构的误差较大，实际上这是由于取样信号的混叠所产生的，也就是说，这时不能从 x(n)中恢复原信号 $x_a(t)$，重构信号如图 3.16 所示。

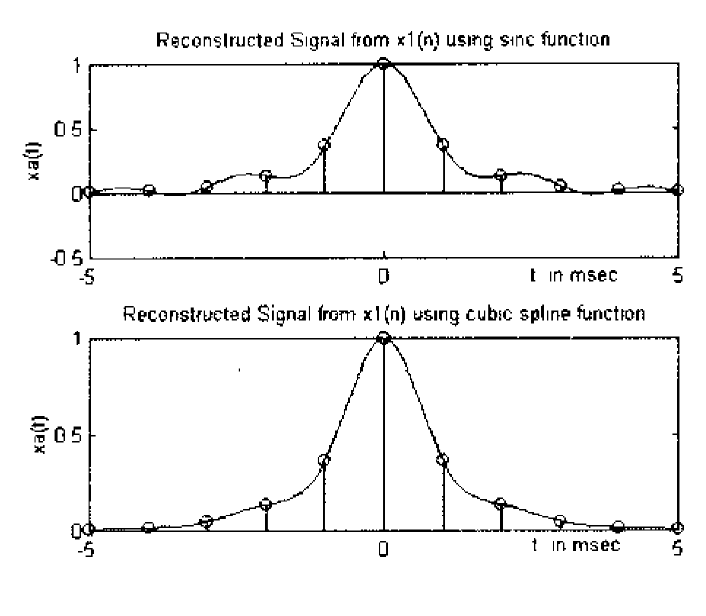


图 3.16　信号重构

3.3　Z 变换

3.3.1　双边 Z 变换

例 3.16　双边序列 $x(n)=a^n u(n)-b^n u(-n-1)$，求其 Z 变换 X(z)，并确定 ROC。

解　记 $x_1(n)=a^n u(n)$，$x_2(n)=-b^n u(-n-1)$，它们分别为正时间序列和负时间序列。$x(n)=x_1(n)+x_2(n)$，下面先求出序列 $x_1(n)$和 $x_2(n)$的 Z 变换。

(1) $x_1(n)=a^n u(n)$

$$X_1(z)=\sum_{n=0}^{\infty}a^n z^{-n}=\sum_{n=0}^{\infty}\left(\frac{a}{z}\right)^n=\frac{1}{1-az^{-1}}\qquad \left|\frac{a}{z}\right|<1$$

因此，$X_1(z)=\frac{z}{z-a}$　ROC_1：$|a|<|z|<\infty$

(2) $x_2(n)=-b^n u(-n-1)$

$$X_2(z) = -\sum_{n=-\infty}^{-1} b^n z^{-n} = -\sum_{n=1}^{\infty}\left(\frac{z}{b}\right)^n = 1 - \sum_{n=0}^{\infty}\left(\frac{z}{b}\right)^n$$

$$= 1 - \frac{1}{1 - z/b} = \frac{z}{z - b} \qquad \mathrm{ROC}_2: 0 < |z| < |b|$$

(3) $x(n)=x_1(n)+x_2(n)$

$$X(z) = X_1(z) + X_2(z) = \frac{z}{z-a} + \frac{z}{z-b} \quad \mathrm{ROC}: \mathrm{ROC}_1 \cap \mathrm{ROC}_2$$

因此当$|a|<|b|$时，ROC：$|a|<|z|<|b|$；当$|a|>|b|$时，ROC为空即$X(z)$不存在。

3.3.2 Z变换重要特性

例 3.17 设$X_1(z)=z+2+3z^{-1}$，$X_2(z)=2z^2+4z+3+5z^{-1}$，求$X_3(z)=X_1(z)X_2(z)$。

解 由Z变换定义可得

$$x_1(n) = \{1, 2, 3\} \quad n = \{-1, 0, 1\}$$

$$x_2(n) = \{2, 4, 3, 5\} \quad n = \{-2, -1, 0, 1\}$$

因此$x_3(n)=x_1(n)\circledast x_2(n)$。

MATLAB程序为ex3017.m：

```
% Example 3.17
%
x1=[1, 2, 3]; n1=-1:1;
x2=[2, 4, 3, 5]; n2=-2:1;
[x3, n3]=conv_m(x1, n1, x2, n2)
```

执行结果为

```
x3=
    2    8    17   23   19   15
n3=
   -3   -2   -1    0    1    2
```

因此得到

$$X_3(z) = 2z^3 + 8z^2 + 17z + 23 + 19z^{-1} + 15z^{-2}$$

3.3.3 逆Z变换

例 3.18 计算

$$X(z) = \frac{1}{(1-0.9z^{-1})^2(1+0.9z^{-1})} \qquad |z| > 0.9$$

的逆Z变换。

解 MATLAB程序为ex3018.m：

```
% Example 3.18
%
b=1;a=poly([0.9 0.9 -0.9]);
[R, P, C]=residuez(b, a)
```

执行结果为

```
R=
  0.2500
  0.5000
  0.2500
P=
  0.9000
  0.9000
  -0.9000
C=
  [ ]
```

因此得到

$$X(z)=\frac{0.25}{1-0.9z^{-1}}+\frac{0.5}{(1-0.9z^{-1})^2}+\frac{0.25}{1+0.9z^{-1}}\quad |z|>0.9$$

相应的逆 Z 变换为

$$\begin{aligned}x(n)&=0.25(0.9)^n u(n)+\frac{5}{9}(n+1)(0.9)^{n-1}u(n+1)+0.25(-0.9)^n u(n)\\&=0.75(0.9)^n u(n)+0.5n(0.9)^{n+1}u(n)+0.25(-0.9)^n u(n)\end{aligned}$$

3.3.4 Z 域系统表示

例 3.19 一因果 LTI 系统

$$y(n)=0.81y(n-2)+x(n)-x(n-2)$$

求(1) $H(z)$；(2) 冲激响应 $h(n)$；(3) 单位阶跃响应 $u(n)$；(4) $H(e^{j\omega})$，并绘出幅频和相频特性。

解 (1) 由差分方程直接得

$$H(z)=\frac{1-z^{-2}}{1-0.81z^{-2}}=\frac{1-z^{-2}}{(1+0.9z^{-1})(1-0.9z^{-1})}\qquad |z|>0.9$$

(2) (3) (4)的 MATLAB 程序为 ex3019.m：

```
% Example 3.19
%
% part 2
b=[1, 0, -1]; a=[1, 0, -0.81];
figure(1)
subplot(2, 1, 1)
dimpulse(b, a, 50)
gtext('impulse response')
% part 3
subplot(2, 1, 2)
dstep(b, a, 50)
gtext('step response')
% part 4
```

```
figure(2)
w=[0:1:500]*pi/500;
freqz(b, a, w);
```

执行后可得到如图 3.17、3.18 所示的图形。图 3.17 中显示出系统的冲激响应和阶跃响应，图 3.18 显示出系统的幅频特性和相频特性。

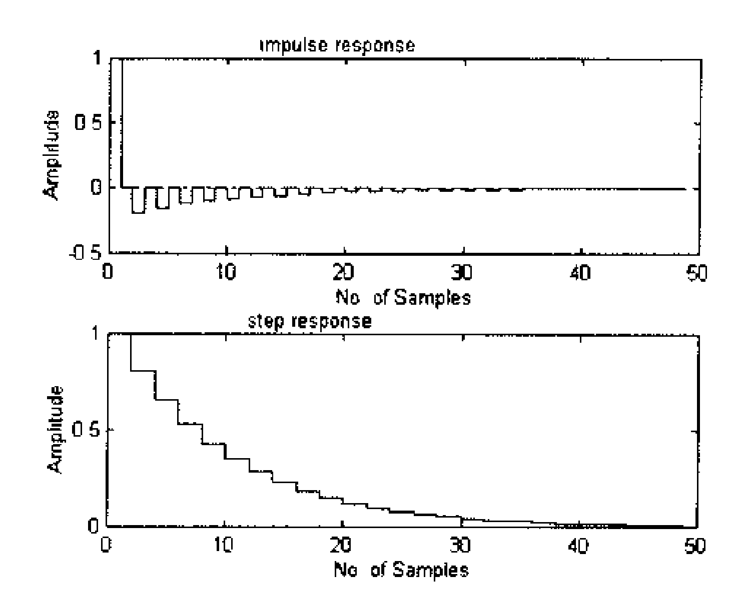


图 3.17　系统的冲激响应和阶跃响应

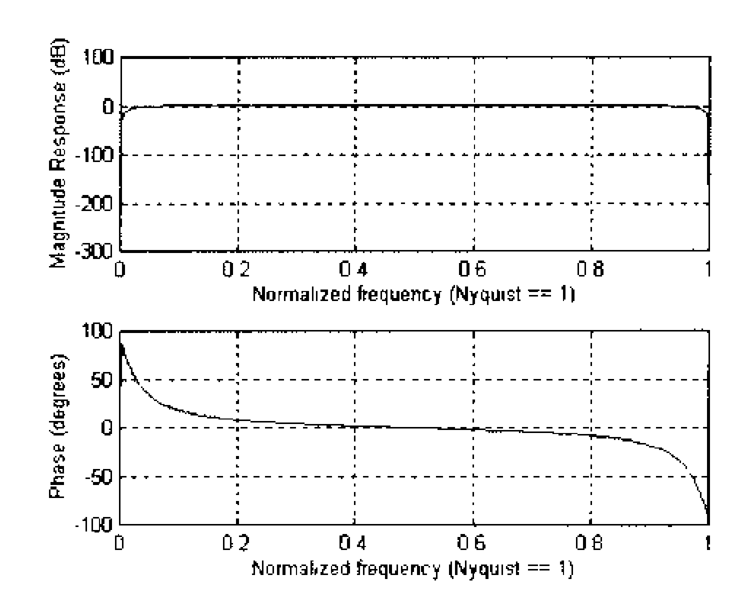


图 3.18　系统的幅频和相频特性

3.3.5 差分方程求解

例 3.20　求解差分方程

$$y(n)=\frac{1}{3}[x(n)+x(n-1)+x(n-2)]+0.95y(n-1)-0.9025y(n-2)$$

$$n\geqslant 0$$

其中 $x(n)=\cos(\pi n/3)$，$y(-1)=-2$，$y(-2)=-3$，$x(-1)=1$，$x(-2)=1$。

解 重写差分方程

$$y(n)-0.95y(n-1)+0.9025y(n-2)=\frac{1}{3}[x(n)+x(n-1)+x(n-2)] \qquad n\geqslant 0$$

因此 MATLAB 程序为 ex3020.m：

```
% Example 3.20
%
b=[1, 1, 1]/3; a=[1, -0.95, 0.9025];
Y=[-2, -3]; X=[1, 1];                          %initial
xic=filtic(b, a, Y, X)
bxplus=[1, -0.5]; axplus=[1, -1, 1];           % X(z)
ayplus=conv(a, axplus)
byplus=conv(b, bxplus)+conv(xic, axplus)
[R, p, C]=residuez(byplus, ayplus)
Mp=abs(p), Ap=angle(p)/pi
%
% plot
%
n=[0:50];
x=cos(pi*n/3);
y=filter(b, a, x, xic);
plot(n, y)
```

执行结果为

```
xic =
     1.4742        2.1383
ayplus =
     1.0000       -1.9500       2.8525      -1.8525      0.9025
byplus =
     1.8075        0.8308      -0.4975       1.9717
R =
     0.0584 - 3.9468i
     0.0584 + 3.9468i
     0.8453 + 2.0311i
     0.8453 - 2.0311i
p =
     0.5000 + 0.8660i
     0.5000 - 0.8660i
     0.4750 + 0.8227i
     0.4750 - 0.8227i
```

```
C =
     [ ]
Mp =
     1.0000
     1.0000
     0.9500
     0.9500
Ap =
     0.3333
     -0.3333
     0.3333
     -0.3333
```

取 50 个输入样本，其响应曲线如图 3.19 所示。

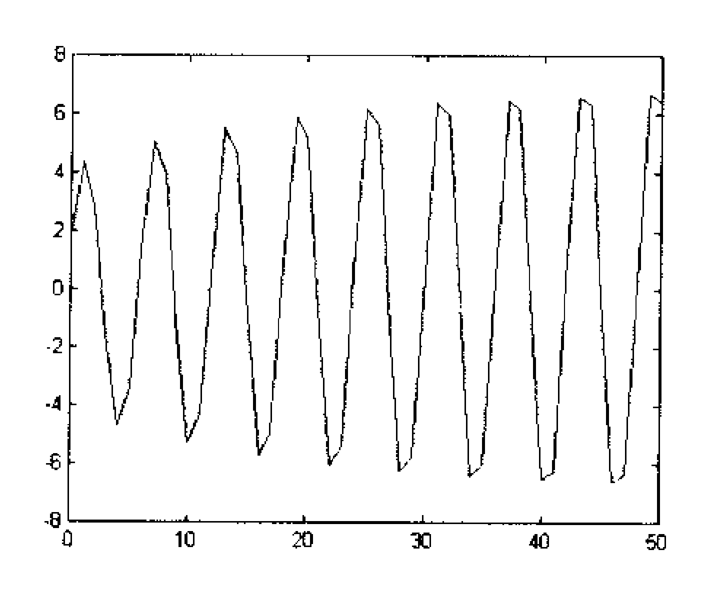


图 3.19 系统的完全响应曲线

3.4 离散傅里叶变换

MATLAB 的扩展函数有 9 种。

1. 计算离散傅里叶级数系数

```
dfs.m
function [Xk]=dfs(xn, N)
% Computes Discrete Fourier Series Coefficients
% ------------------
```

```
% [Xk]=dfs(xn, N)
% Xk=DFS coeff. array over 0<=k<N-1
% xn=One period of periodic signal over 0<=n<=N-1
% N=Fundamental period of xn
%
N=[0:1:N-1];
k=[0:1:N-1];
WN=exp(-j * 2 * pi/N);          % Wn factor
nk=n' * k;
WNnk=WN. ^ nk;                  % DFS matrix
Xk=xn * WNnk;                   % row vector for DFS coefficients
```

2. 计算逆离散傅里叶级数系数

```
idfs. m
function [Xn]=idfs(Xk, N)
% Computes Inverse Discrete Fourier Series
% ------------------
% [xn]=idfs(Xk, N)
% xn=One period of periodic signal over 0<=n<=N-1
% Xk=DFS coeff. array over 0<=k<=N-1
% N=Fundamental period of Xk
%
N=[0:1:N-1];
k=[0:1:N-1];
WN=exp(-j * 2 * pi/N);          % Wn factor
nk=n' * k;
WNnk=WN. ^ nk;                  % IDFS matrix
xn=(Xk * WNnk)/N;               % row vector for IDFS coefficients
```

3. 取余运算

MATLAB 自身提供的 rem 函数可计算余数

```
m=rem(n, N)
```

这时 $n \geqslant 0$，m 为 n 除 N 后得到的余数。当 $n<0$ 时，rem 不能采用，故我们提供 n 为任意值时求余数的函数 mod。

```
mod. m
function m=mod(n, N)
% Computes m=(n mod N) index
% ------------------
```

```
% m=m(n, N)
m=rem(n, N);
m=m+N;
m=rem(m, N);
```

4. 离散傅里叶变换

```
dft.m
function [Xk]=dft(xn, N)
% Computes Discrete Fourier Transform
% -----------------
% [Xk]=dft(xn, N)
% Xk=DFT coeff. array over 0<=k<=N-1
% xn=N-point finite-duration sequence
% N=Length of DFT
%
n=[0:1:N-1];
k=[0:1:N-1];
WN=exp(-j*2*pi/N);          % Wn factor
nk=n'*k;
WNnk=WN.^nk;                % DFT matrix
Xk=xn*WNnk;                 % row vector for DFT coefficients
```

5. 逆离散傅里叶变换

```
idft.m
function [xn]=idft(Xk, N)
% Computes Inverse Discrete Fourier Transform
% -----------------
% [xn]=idft(Xk, N)
% xn=N-point sequence over 0<=n<=N-1
% Xk=DFT coeff. array over 0<=k<=N-1
% N=Length of DFT
%
N=[0:1:N-1];
k=[0:1:N-1];
WN=exp(-j*2*pi/N);          % Wn factor
nk=n'*k;
WNnk=WN.^(-nk);             % IDFT matrix
xn=(Xk*WNnk)/N;             % row vector for IDFT values
```

6. 实序列的奇偶分解

```
circevod. m
function [xec, xoc]=circevod(x)
% signal decomposition into circular - even and circular - odd parts
% -----------------
% [xec, xoc]=circecod(x)
%
if any (imag(x)=0)
      error('x is not a real sequence')
end
N=length(x); n=0:(N-1);
xec=0.5 * (x+x(mod(-n, N)+1));
xoc=0.5 * (x-x * mod(-n, N)+1));
```

7. 序列的循环移位

```
cirshftt. m
function y=cirshftt(x, m, N)
% Circular shift of m samples wrt size N in sequence x:(time domain)
% -----------------
% [y]=cirshftt(x, m, N)
% y=output sequence containing the circular shift
% x=input sequence of length <=N
% m=sample shift
% N=size of circular buffer
% Method: y(n)=x((n-m) mod N)

% Check for length of x
if length(x)>N
      error('N must be >= the length of x')
end
x=[x zeros(1, N-length(x))];
n=[0:1:N-1];
n=mod(n-m, N);
y=x(n+1);
```

8. 序列的循环卷积

```
circonvt. m
```

```
function y=circonvt(x1, x2, N)
% N-point circular convolution between x1 and x2:(time-domain)
% -----------------
% [y]=circonvt(x1, x2, N)
% y=output sequence containing the circular convolution
% x1=input sequence of length N1<=N
% x2=input sequence of length N2<=N
% N=size of circular buffer
% Method: y(n)=sum(x1(m)*x2((n-m) mod N))

% Check for length of x1
if length(x1)>N
      error('N must be >= the length of x1')
end

% Check for length of x2
if length(x2)>N
      error('N must be >= the length of x2')
end

x1=[x1 zeros(1, N-length(x1))];
x2=[x2 zeros(1, N-length(x2))];
m=[0:1:N-1];
x2=x2(mod(-m, N)+1);
H=zeros(N, N);
for n=1:1:N
      H(n, :)=cirshftt(x2, n-1, N);
end
y=x1*H';
```

9. 分块卷积

长序列的卷积可通过重叠保存方法进行分块，然后经 DFT 得到。

```
ovrlpsav.m
function [y]=ovrlpsav(x, h, N)
% Overlap-Save method of block convolution
% -----------------
% [y]=ovrlpsav(x, h, N)
% y=output sequence
% x=input sequence
```

```
% h=impulse response
% N=block length
%
Lenx=length(x); M=length(h);
M1=M-1; L=N-M1;
h=[h zeros(1, N-M)];
%
x=[zeros(1, M1), x, zeros(1, N-1)];        % preappend (M-1) zeros
K=floor(Lenx+M1-1)/(L));
Y=zeros(K+1, N);
% convolution with succesive blocks
for k=0:K
        xk=x(k*L+1:k*L+N);
        Y(k+1, :)=circonvt(xk, h, N);
end
Y=Y(:, M:N)';                              % discard the first (M-1) samples
y=(Y(:))';                                 % assemble output
```

3.4.1 离散傅里叶级数

例 3.21 周期方波序列

$$\tilde{x}(n)=\begin{cases}1 & mN\leqslant n\leqslant mN+L-1\\ 0 & mN+L\leqslant n\leqslant (m+1)N-1\end{cases}\quad m=0,\pm 1,\cdots$$

绘制出下列情况下的 $\tilde{X}(k)$的幅度：

(1) L=5，N=20

(2) L=5，N=40

(3) L=5，N=60

(4) L=7，N=60

解 利用扩展函数 dfs 可直接求解，MATLAB 程序为 ex3021.m：

```
% Example 3.21
%
% Part 1
L=5; N=20;
n=1:N;
xn=[ones(1, L), zeros(1, N-L)];
Xk=dfs(xn, N);
magXk=abs([Xk(N/2+1:N) Xk(1:N/2+1)]);
k=[-N/2:N/2];
figure(1)
subplot(2, 1, 1)
```

```
stem(n, xn); xlabel('n'); ylabel('xtilde(n)')
title('DFS of SQ. wave: L=5, N=20')
subplot(2, 1, 2); stem(k, magXk); axis([-N/2, N/2, -0.5, 5.5])
xlabel('k'); ylabel('Xtilde(k)')

% Part 2
L=5; N=40;
n=1:N;
xn=[ones(1, L), zeros(1, N-L)];
Xk=dfs(xn, N);
magXk=abs([Xk(N/2+1:N) Xk(1:N/2+1)]);
k=[-N/2:N/2];
figure(2)
subplot(2, 1, 1)
stem(n, xn); xlabel('n'); ylabel('xtilde(n)')
title('DFS of SQ. wave: L=5, N=40')
subplot(2, 1, 2); stem(k, magXk); axis([-N/2, N/2, -0.5, 5.5])
xlabel('k'); ylabel('Xtilde(k)')

% Part 3
L=5; N=60;
n=1:N;
xn=[ones(1, L), zeros(1, N-L)];
Xk=dfs(xn, N);
magXk=abs([Xk(N/2+1:N) Xk(1:N/2+1)]);
k=[-N/2:N/2];
figure(3)
subplot(2, 1, 1)
stem(n, xn); xlabel('n'); ylabel('xtilde(n)')
title('DFS of SQ. wave: L=5, N=60')
subplot(2, 1, 2); stem(k, magXk); axis([-N/2, N/2, -0.5, 5.5])
xlabel('k'); ylabel('Xtilde(k)')

% Part 4
L=7; N=60;
n=1:N;
xn=[ones(1, L), zeros(1, N-L)];
Xk=dfs(xn, N);
magXk=abs([Xk(N/2+1:N) Xk(1:N/2+1)]);
```

```
k=[-N/2:N/2];
figure(4)
subplot(2, 1, 1)
stem(n, xn); xlabel('n'); ylabel('xtilde(n)')
title('DFS of SQ. wave:L=7, N=60')
subplot(2, 1, 2); stem(k, magXk); axis([-N/2, N/2, -0.5, 7.5])
xlabel('k'); ylabel('Xtilde(k)')
```

执行程序可得如图 3.20～3.23 所示的曲线。每个图中，上图为周期序列 $\tilde{x}(n)$，下图为相应的 $\tilde{X}(k)$。

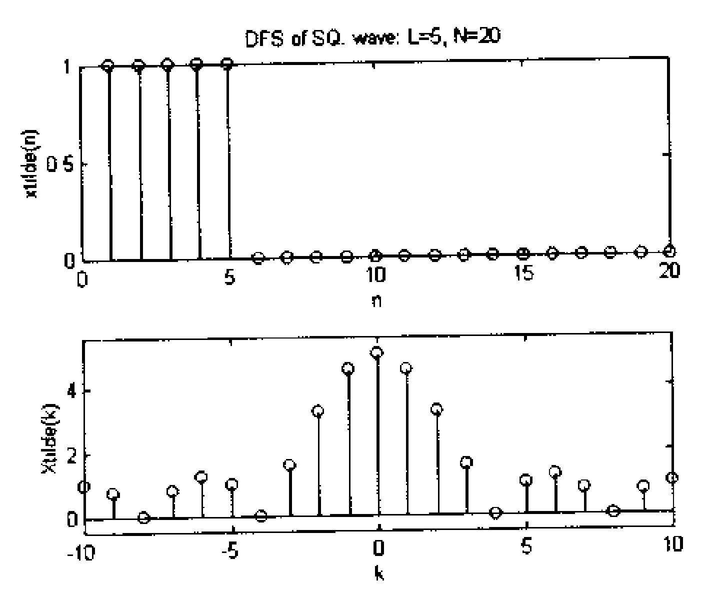


图 3.20 $\tilde{x}(n)$和 $\tilde{X}(k)$(L=5, N=20)

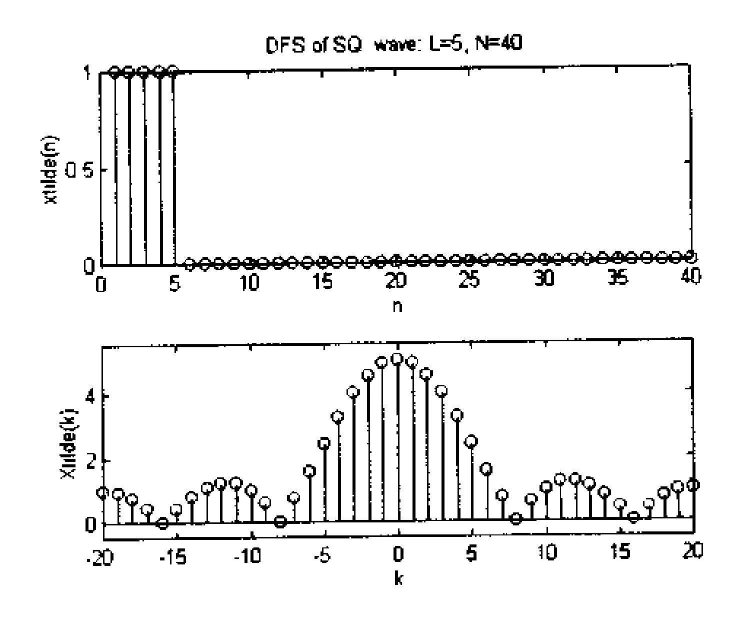


图 3.21 $\tilde{x}(n)$和 $\tilde{X}(k)$(L=5, N=40)

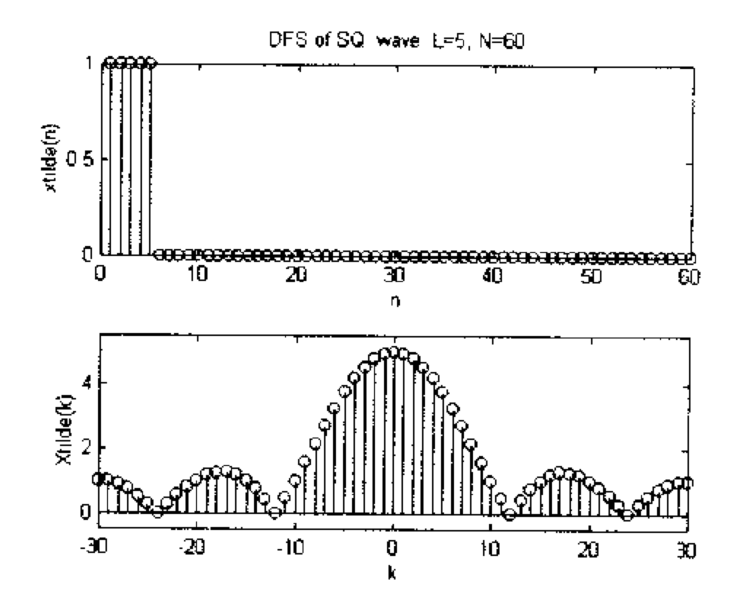


图 3.22 $\tilde{x}(n)$和$\tilde{X}(k)(L=5, N=60)$

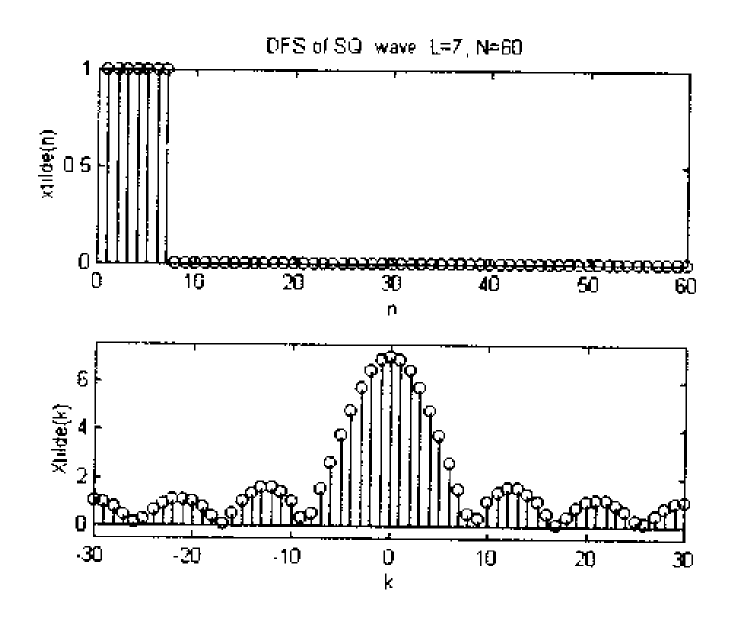


图 3.23 $\tilde{x}(n)$和$\tilde{X}(k)(L=7, N=60)$

例 3.22 设$x(n)=(0.7)^n u(n)$，在单位圆上以$N=5$和$N=20$对其Z变换取样，并研究时域信号的影响。

解 由常见序列的Z变换得

$$X(z)=\frac{1}{1-0.7z^{-1}}=\frac{z}{z-0.7} \qquad |z|>0.7$$

由此可计算出

$$\tilde{X}(k)=X(z)|_{z=e^{j2\pi k/N}} \qquad k=0, \pm 1, \pm 2, \cdots$$

最后利用IDFS计算相应的时序列$\tilde{x}(n)$。

MATLAB 程序为 ex3022.m：

```
% Example 3.22
```

```
%
N=5;
k=0:1:N-1;
wk=2*pi*k/N;
zk=exp(j*wk);
Xk=(zk)./(zk-0.7);
xn=real(idfs(Xk, N));
xtilde=xn' * ones(1, 8); xtilde=(xtilde(:))';
subplot(2, 1, 1); stem(0:39, xtilde); axis([0, 40, -0.1, 1.5])
xlabel('n'); ylabel('xtilde(n)'); gtext('N=5')

%
N=20;
k=0:1:N-1;
wk=2*pi*k/N;
zk=exp(j*wk);
Xk=(zk)./(zk-0.7);
xn=real(idfs(Xk, N));
xtilde=xn' * ones(1, 2); xtilde=(xtilde(:))';
subplot(2, 1, 2); stem(0:39, xtilde); axis([0, 40, -0.1, 1.5])
xlabel('n'); ylabel('xtilde(n)'); gtext('N=20')
```

执行后得到如图 3.24 所示的周期序列。由图不难看出，当 N=5 时，序列发生了重叠现象，而在 N=20 时，序列之间无重叠现象。

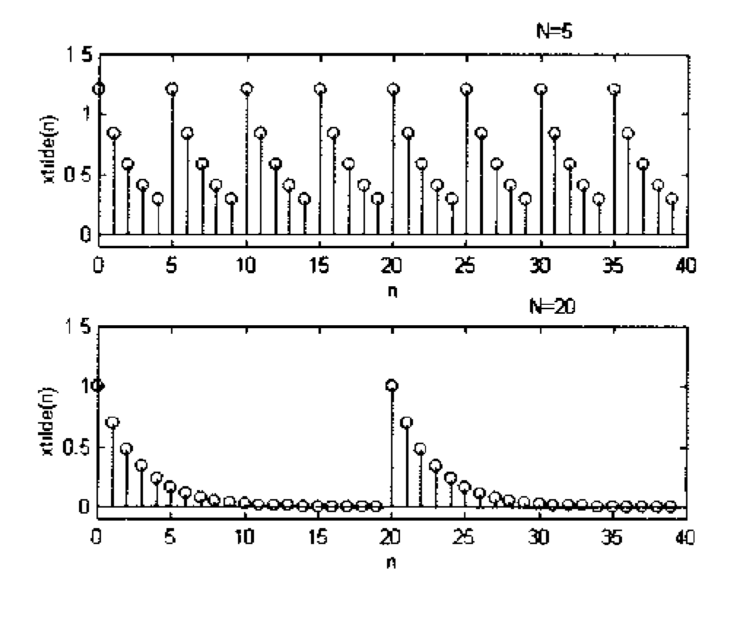


图 3.24　周期序列

3.4.2 离散傅里叶变换

例 3.23 为说明高密度频谱和高分辨频谱之间的差异，考虑

$$x(n)=\cos(0.48\pi n)+\cos(0.52\pi n)$$

(1) 取 $x(n)(0\leqslant n\leqslant 10)$时，求 $x(n)$的 DFT $X(k)$；

(2) 将(1)中的 $x(n)$以补零方式使 $x(n)$加长到$0\leqslant n\leqslant 100$，求 $X(k)$；

(3) 取 $x(n)(0\leqslant n\leqslant 100)$，求 $X(k)$。

解 MATLAB 程序为 ex3023.m：

```
% Example 3.23
%
% part 1
% Spectrum based on the first 10 samples of x(n)
figure(1)
n1=[0:1:9]; y1=x(1:1:10);
subplot(2, 1, 1); stem(n1, y1); title('signal x(n), 0 <= n <= 9'); xlabel('n')
axis([0, 10, -2.5, 2.5])
Y1=fft(y1); magY1=abs(Y1(1:1:6));
k1=0:1:5; w1=2*pi/10*k1;
subplot(2, 1, 2); stem(w1/pi, magY1); title('Samples of DTFT Magnitude');
xlabel('frequency in pi units')
axis([0, 1, 0, 10])

% part 2
% High density spectrum (100 samples) based on the first 10 samples of x(n)
figure(2)
n3=[0:1:99]; y3=[x(1:1:10) zeros(1, 90)];
subplot(2, 1, 1); stem(n3, y3); title('signal x(n), 0 <= n <= 9 + 90 zeros');
xlabel('n')
axis([0, 100, -2.5, 2.5])
Y3=fft(y3); magY3=abs(Y3(1:1:51));
k3=0:1:50; w3=2*pi/100*k3;
subplot(2, 1, 2); plot(w3/pi, magY3); title('DTFT Magnitude');
xlabel('frequency in pi units')
axis([0, 1, 0, 10])

% part 3
% High resolution spectrum based on 100 samples of the signal x(n)
figure(3)
n=[0:1:99];
```

```
x=cos(0.48 * pi * n)+cos(0.52 * pi * n);
subplot(2, 1, 1); stem(n, x); title('signal x(n), 0 <= n <= 99'); xlabel('n')
axis([0, 100, -2.5, 2.5])
X=fft(x); magX=abs(X(1:1:51));
k=0:1:50; w=2 * pi/100 * k;
subplot(2, 1, 2); plot(w/pi, magX); title('DTFT Magnitude');
xlabel('frequency in pi units')
axis([0, 1, 0, 60])
```

执行后可得到如图 3.25～3.27 所示的图形。图 3.25 为$0 \leqslant n \leqslant 10$ 时的序列 x(n)和相应的 DFT X(k)，从图中几乎无法看出有关信号频谱的信息。图 3.26 是将 x(n)补 90 个零时的 x(n)和相应的 X(k)，显然，这时的谱线相当密，故称为高密度频谱图，但从图中很难看出信号的频谱部分。图 3.27 为加长取样数据长度，$0 \leqslant n \leqslant 100$，这时可很清晰地看出信号的频谱成分，这称为高分辨频谱。

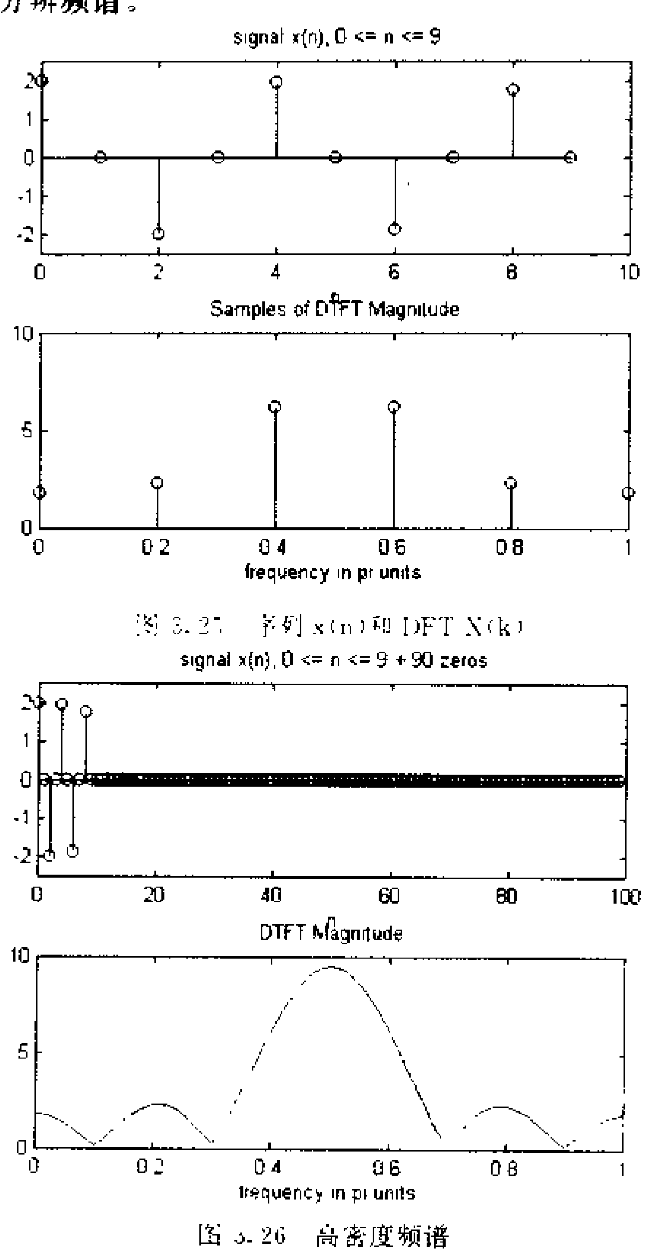


图 3.25　序列 x(n)和 DFT X(k)

图 3.26　高密度频谱

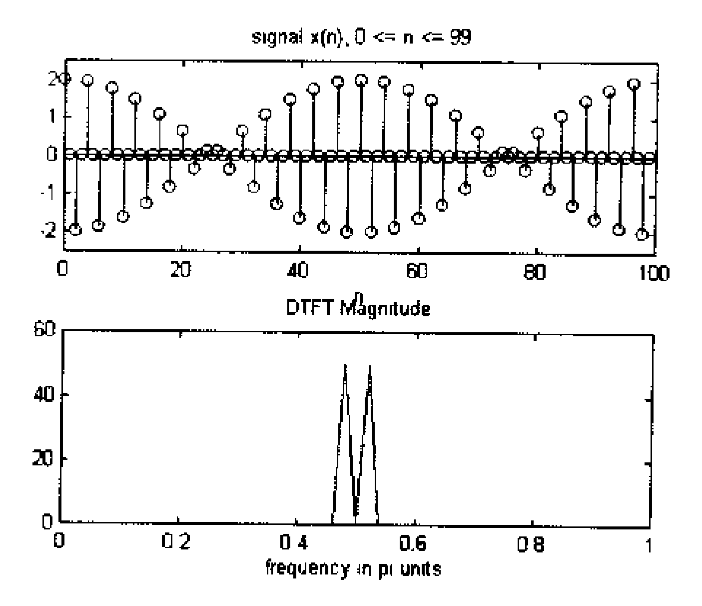


图 3.27 高分辨频谱

例 3.24 设 $x(n)=10(0.8)^n \quad 0\leqslant n\leqslant 10$

(1) 分解 $x(n)$ 成 $x_{ec}(n)$ 和 $x_{oc}(n)$（偶、奇部分）；

(2) 检验实序列的性质：

$$DFT[x_{ec}(n)]=Re[X(k)]$$

$$DFT[x_{oc}(n)]=Im[X(k)]$$

解 MATLAB 程序为 ex3024.m：

```
% Example 3.24
%

% part 1:plot xec(n) and xoc(n)
figure(1)
n=0:10; x=10*(0.8).^ n;
[xec, xoc]=circevod(x);
subplot(2, 1, 1); stem(n, xec); title('Circular-even component')
xlabel('n'); ylabel('xec(n)'); axis([-0.5, 10.5, -1, 11])
subplot(2, 1, 2); stem(n, xoc); title('Circular-odd component')
xlabel('n'); ylabel('xoc(n)'); axis([-0.5, 10.5, -4, 4])

% part 2:verify property
%
figure(2)
X=dft(x, 11); Xec=dft(xec, 11); Xoc=dft(xoc, 11);
subplot(2, 2, 1); stem(n, real(X)); axis([-0.5, 10.5, -5, 50])
title('Real{DFT[x(n)]}'); xlabel('k');
```

```
subplot(2, 2, 2); stem(n, imag(X)); axis([-0.5, 10.5, -20, 20])
title('Imag{DFT[x(n)]}'); xlabel('k');
subplot(2, 2, 3); stem(n, real(Xec)); axis([-0.5, 10.5, -5, 50])
title('DFT[xec(n)]'); xlabel('k');
subplot(2, 2, 4); stem(n, imag(Xoc)); axis([-0.5, 10.5, -20, 20])
title('DFT[xoc(n)]'); xlabel('k');
```

执行后可得如图 3.28～3.29 所示的曲线。图 3.28 为序列 x(n)的奇偶分量。图 3.29为 x(n)的 DFT 实部、虚部分量及 $x_{ec}(n)$和 $x_{oc}(n)$的 DFT，从图中不难看出：

$$\mathrm{DFT}[x_{ec}(n)]=\mathrm{Re}[X(k)]$$

$$\mathrm{DFT}[x_{oc}(n)]=\mathrm{Im}[X(k)]$$

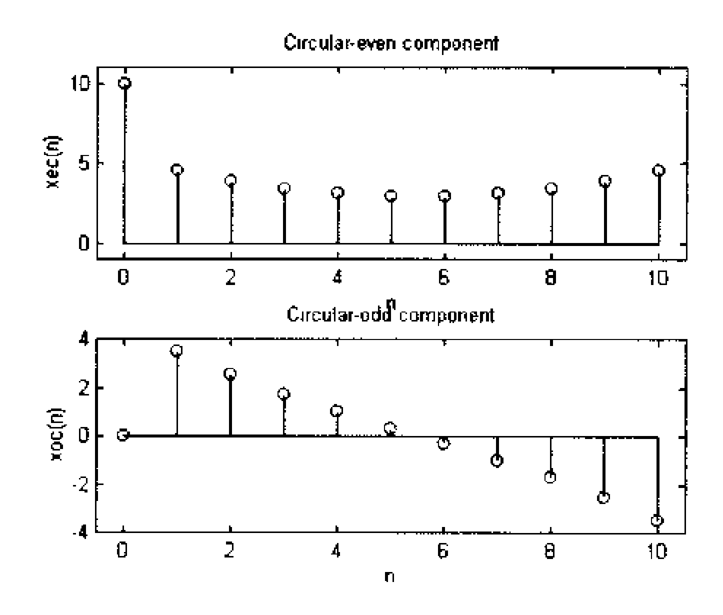


图 3.28 $x_{ec}(n)$和 $x_{oc}(n)$

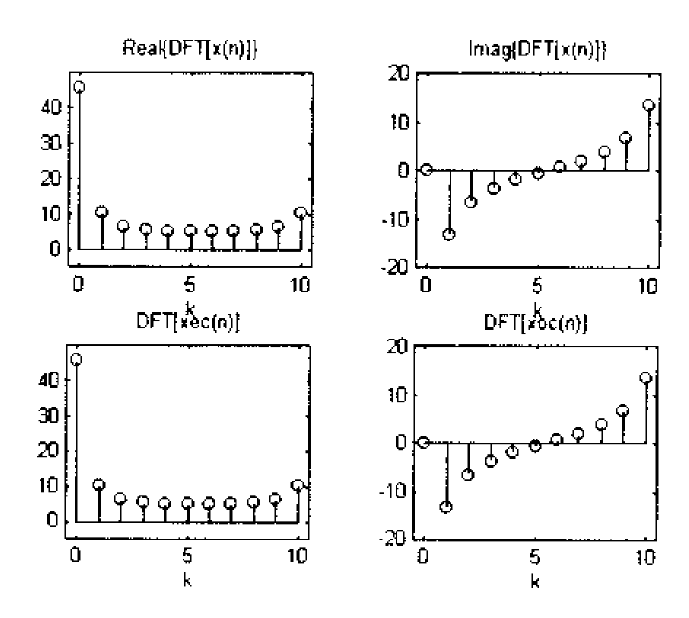


图 3.29 X(k)与 $X_{ec}(k)$和 $X_{oc}(k)$的关系

例 3.25 设 $x(n)=10(0.8)^n \quad 0 \leqslant n \leqslant 10$。

求 $y(n)=x((n-6))_{15}$

解 可直接采用循环移位函数 cirshftt 进行求解，MATLAB 程序为 ex3025.m：

```
% Example 3.25
%
% plot x((n-6))15
figure(1)
n=0:10; x=10*(0.8).^ n;
y=cirshftt(x, 6, 15);
n=0:14; x=[x, zeros(1, 4)];
subplot(2, 1, 1); stem(n, x); title('Original sequence')
xlabel('n'); ylabel('x(n)'); axis([-1, 15, -1, 11])
subplot(2, 1, 2); stem(n, y);
title('Circularly shifted sequence, N=15')
xlabel('n'); ylabel('x((n-6) mod 15)');
axis([-1, 15, -1, 11])
```

执行后可得如图 3.30 所示的曲线。

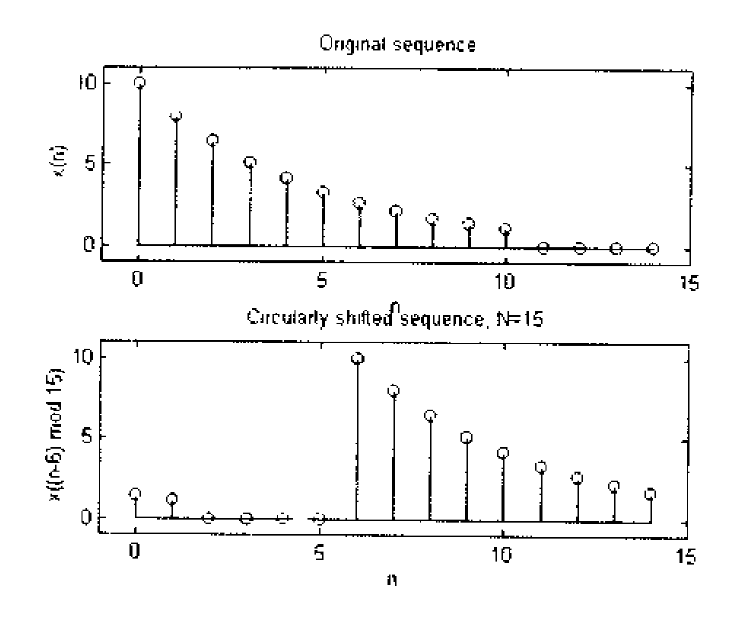


图 3.30 序列的循环移位

例 3.26 设 $x_1(n)=\{1, 2, 2\}$，$x_2(n)=\{1, 2, 5, 4\}$，试分别计算

(1) $y_1(n)=x_1(n)⑤x_2(n)$

(2) $y_2(n)=x_1(n)⑥x_2(n)$

解 可直接利用循环卷积 circonvt 函数计算，MATLAB 程序为 ex3026.m：

```
% Example 3.26
```

```
%
x1=[1, 2, 2]; x2=[1, 2, 5, 4];
y1=circonvt(x1, x2, 5)
y2=circonvt(x1, x2, 6)
```

执行后即可得到

```
y1=
  9  4  11  18  18
y2=
  1  4  11  18  18  8
```

3.4.3 利用 DFT 计算线性卷积

例 3.27 令 $x(n)=n+1$，$0\leqslant n\leqslant 9$，$h(n)=\{1,0,-1\}$，利用 $N=6$ 的重复法求 $y(n)=x(n)\circledast h(n)$。

解 $M=3$，$N=6$，因此分块时每块长度为 6，重复 $M-1=2$ 个取样。

$$x(n)=\{1,2,3,4,5,6,7,8,9,10\}$$

分块后

$$x_1(n)=\{0,0,1,2,3,4\}$$
$$x_2(n)=\{3,4,5,6,7,8\}$$
$$x_3(n)=\{7,8,9,10,0,0\}$$

这样可分别计算

$$y_1=x_1(n)⑥h(n)=\{-3,-4,1,2,2,2\}$$
$$y_2=x_2(n)⑥h(n)=\{-4,-4,2,2,2,2\}$$
$$y_3=x_3(n)⑥h(n)=\{7,8,2,2,-9,-10\}$$

注意，前面两个取样值应该舍弃，这样就得到了 $y(n)$

$$y(n)=\{1,2,2,2,2,2,2,2,2,2,-9,-10\}$$

这与线性卷积

$$y(n)=x(n)\circledast h(n)$$

的结果一致。

这在 MATLAB 中可很容易实现。如果借助 ovrlpsav 函数就更加方便

```
n=0:9; x=n+1; h=[1, 0, -1]; N=6;
y=ovrlpsav(x, h, N)
```

执行后得

```
y=
  1  2  2  2  2  2  2  2  2  2  -9  -10
```

3.4.4 快速傅里叶变换(FFT)

例 3.28 模拟信号 $x(t)=2\sin(4\pi t)+5\cos(8\pi t)$，以 $t=0.01n(n=0:N-1)$进行取样，求 N 点 DFT 的幅值谱。N 分别为(1) $N=45$；(2) $N=50$；(3) $N=55$；(4) $N=60$

解 MATLAB 程序为 ex3028.m：

```
% Example 3.28
%
figure(1)
subplot(2, 2, 1)
N=45; n=0:N-1; t=0.01*n;
q=n*2*pi/N;
x=2*sin(4*pi*t)+5*cos(8*pi*t);
y=fft(x, N);
plot(q, abs(y))
title('FFT N=45')
%
subplot(2, 2, 2)
N=50; n=0:N-1; t=0.01*n;
q=n*2*pi/N;
x=2*sin(4*pi*t)+5*cos(8*pi*t);
y=fft(x, N);
plot(q, abs(y))
title('FFT N=50')
%
subplot(2, 2, 3)
N=55; n=0:N-1; t=0.01*n;
q=n*2*pi/N;
x=2*sin(4*pi*t)+5*cos(8*pi*t);
y=fft(x, N);
plot(q, abs(y))
title('FFT N=55')
%
subplot(2, 2, 4)
N=60; n=0:N-1; t=0.01*n;
q=n*2*pi/N;
x=2*sin(4*pi*t)+5*cos(8*pi*t);
y=fft(x, N);
plot(q, abs(y))
title('FFT N=60')
```

执行后可得如图 3.31 所示的信号谱。从图中可以看出，这几种情况下均有较好的精度。

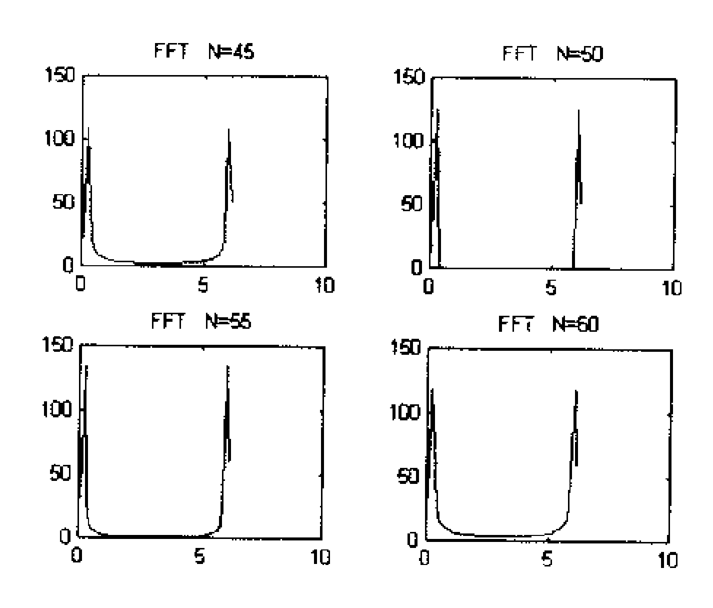


图 3.31 x(t)的信号谱

例 3.29 在上例的基础上，N=64，并在信号中加入噪声(正态)w(t)

$$x(t) = 2\sin(4\pi t) + 5\cos(8\pi t) + 0.8w(t)$$

试比较有无噪声时的信号谱。

解 MATLAB 程序为 ex3029.m：

```
% Example 3.29
%
figure(1)
subplot(2, 1, 1)
N=64; n=0:N-1; t=0.01*n;
q=n*2*pi/N;
x=2*sin(4*pi*t)+5*cos(8*pi*t);
y=fft(x, N);
plot(q, abs(y))
title('FFT N=64')
%
subplot(2, 1, 2)
N=64; n=0:N-1; t=0.01*n;
q=n*2*pi/N;
x=2*sin(4*pi*t)+5*cos(8*pi*t)+0.8*randn(1, N);
y=fft(x, N);
plot(q, abs(y))
title('FFT N=64 (with noise)')
```

执行后可得如图 3.32 所示的信号谱。

从图中可以看出，在信号检测的意义上，这种噪声不会影响信号的检测。

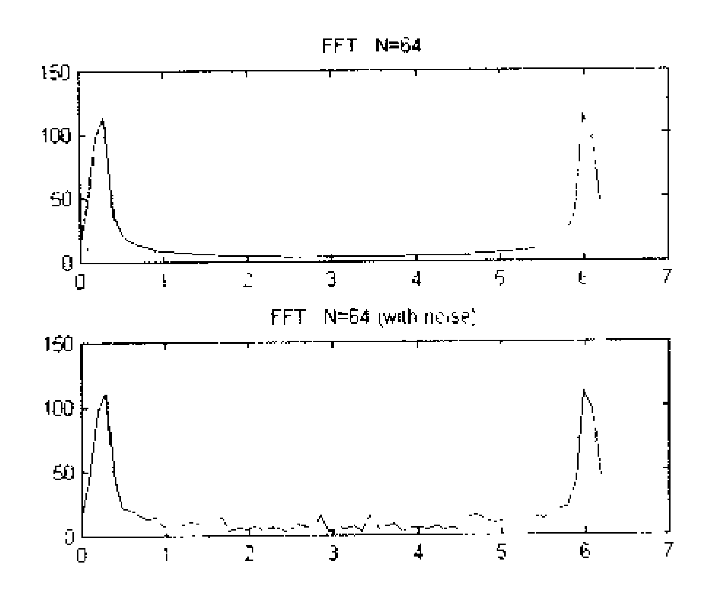


图 3.32 有无噪声时的信号谱

3.5 数字滤波器结构

MATLAB 的扩展函数有11 种。

1. 变直接形式为级联形式

```
dir2cas.m
function [b0, B, A]=dir2cas(b, a);
% DIRECT - form to CASCADE - form conversion (cplxpair version)
% ------------------------------
% [b0, B, A]=dir2cas(b, a)
% b0=gain coefficient
% B=K by 3 matrix of real coefficients containing bk's
% A=K by 3 matrix of real coefficients containing ak's
% b=numberator polynomial coefficients of DIRECT form
% a=denominator polynomial coefficients of DIRECT form

% compute gain coefficient b0
b0=b(1); b=b/b0;
a0=a(1); a=a/a0;
b0=b0/a0;
%
M=length(b); N=length(a);
```

```
if N>M
        b=[b zeros(1, N-M)];
elseif M>N
        a=[a zeros(1, M-N)]; N=M;
else
        NM=0;
end
%
K=floor(N/2); B=zeros(K, 3); A=zeros(K, 3);
if K * 2==N;
        b=[b 0];
        a=[a 0];
end
%
broots=cplxpair(roots(b));
aroots=cplxpair(roots(a));
for i=1:2:2 * K
        Brow=broots(i:1:i+1,:);
        Brow=real(poly(Brow));
        B(fix(i+1)/2,:)=Brow;
        Arow=aroots(i:1:i+1,:);
        Arow=real(poly(Arow));
        A(fix(i+1)/2,:)=Arow;
end
```

2. 滤波器的级联实现

```
casfiltr.m
function y=casfiltr(b0, B, A, x);
% CASCADE form realization of IIR and FIR filters
% ---------------------------
% y=casfiltr(b, A, B, x)
% y=output sequence
% b0=gain coefficient of CASCADE form
% B=K by 3 matrix of real coefficients containing bk's
% A=K by 3 matrix of real coefficients containing ak's
% x=input sequence
%
[K, L]=size(B);
N==length(x);
```

```
w=zeros(K+1, N);
w(1, :)=x;
for i=1:K
        w(i+1, :)=filter(B(i, :), A(i, :), w(i, :));
end
y=b0*w(K+1, :);
```

3. 变级联形式为直接形式

```
cas2dir.m
function [b, a]=cas2dir(b0, B, A);
% CASCADE-to-DIRECT form conversion
% ---------------------------
% [b, a]=cas2dir(b0, B, A)
% b=numerator polynomial coefficients of DIRECT form
% a=denominator polynomial coefficients of DIRECT form
% b0=gain coefficient
% B=K by 3 matrix of real coefficients containing bk's
% A=K by 3 matrix of real coefficients containing ak's
%
[K, L]=size(B);
b=[1];
a=[1];
for i=1:K
        b=conv(b, B(i, :));
        a=conv(a, A(i, :));
end
b=b*b0;
```

4. 变直接形式为并联形式

```
dir2par.m
function [C, B, A]=dir2par(b, a);
% DIRECT-form to PARALLEL-form conversion
% ---------------------------
% [C, B, A]=dir2par(b, a)
% C=Polynomial part when length(b)>=length(a)
% B=K by 3 matrix of real coefficients containing bk's
% A=K by 3 matrix of real coefficients containing ak's
% b=numerator polynomial coefficients of DIRECT form
```

```
% a=denominator polynomial coefficients of DIRECT form
%
M=length(b); N=length(a);
[r1, p1, C]=residuez(b, a);
p=cplxpair(p1, 10000000 * eps);
I=cplxcomp(p1, p);
r=r1(I);
K=floor(N/2); B=zeros(K, 2); A=zeros(K, 3);
if K * 2==N; %N even, order of A(z) odd, one factor is first order
                 for i=1:2:N-2
                     Brow=r(i:1:i+1, :);
                     Arow=p(i:1:i+1, :);
                     [Brow, Arow]=residuez(Brow, Arow, [ ]);
                     B(fix((i+1)/2), :)=real(Brow');
                     A(fix((i+1)/2), :)=real(Arow');
                 end
                 [Brow, Arow]=residuez(r(N-1), p(N-1), [ ]);
                 B(K, :)=[real(Brow') 0]; A(K, :)=[ral(Arow') 0];
else
      for i=1:2:N-1
                     Brow=r(i:1:i+1, :);
                     Arow=p(i:1:i+1, :);
                     [Brow, Arow]=residuez(Brow, Arow, [ ]);
                     B(fix((i+1)/2), :)=real(Brow');
                     A(fix((i+1)/2), :)=real(Arow');
                 end
end
```

5. 复共轭对比较

```
cplxcomp.m
function I=cplxcomp(p1, p2)
% I=cplxcomp(p1, p2)
% Compares two complex pairs which contain the sme scalar elements
% but (possibly) at differrent indices. This routine should be
% used after CPLXPAIR routine for rearranging pole vector and its
% corresponding residue vector.
%        p2=cplxpair(p1)
%
I=[ ];
```

```
for j=1:length(p2)
     for i=1:length(p1)
              if (abs(p1(i)-p2(j))<0.0001)
                I=[I, i];
        end
     end
end
I=I';
```

6. 滤波器的并联实现

```
parfiltr.m
function y=parfiltr(C, B, A, x);
% PARALLEL form realization of IIR filters
% -----------------------------
% [y]=parfiltr(C, B, A, x);
% y=output sequence
% C=polynomial (FIR) part when M>=N
% B=K by 3 matrix of real coefficients containing bk's
% A=K by 3 matrix of real coefficients containing ak's
% x=input sequence
%
[K, L]=size(B);
N=length(x);
w=zeros(K+1, N);
w(1, :)=filter(C, 1, x);
for i=1:K
      w(i+1, :)=filter(B(i, :), A(i, :), x);
end
y=sum(w);
```

7. 变并联形式为直接形式

```
par2dir.m
function [b, a]=par2dir(C, B, A);
% PARALLEL-to-DIRECT form conversion
% -----------------------------
% [b, a]=par2dir(C, A, B)
% b=numerator polynomial coefficients of DIRECT form
% a=denominator polynomial coefficients of DIRECT form
```

```
% C=Polynomial part of PARALLEL form
% B=K by 3 matrix of real coefficients containing bk's
% A=K by 3 matrix of real coefficients containing ak's
%
[K, L]=size(A); R=[ ]; P=[ ];
for i=1:K
            [r, p, k]=residuez(B(i, :), A(i, :));
            R=[R; r]; P=[P; p];
end
[b, a]=residuez(R, P, C);
b=b(:)'; a=a(:)';
```

8. 变 h(n)值形式为频率取样形式

```
dir2fsm.
function [C, B, A]=dir2fs(h);
% Direct form to Frequency Sampling form conversion
% -----------------------------
% [C, B, A]=dirsfs(h)
% C=Row vector containing gains for parallel sections
% B=Matrix containing numerator coefficients arranged in rows
% A=Matrix containing denominator coefficients arranged in rows
% h=impulse response vector of an FIR filter
%
M=length(h);
N=fft(h, M);
magH=abs(H); phaH=angle(H)';
% check even or odd M
if (M==2*floor(M/2))
      L=M/2-1; % M is even
      A1=[1, -1, 0; 1, 1, 0];
      C1=[real(H(1)), real(H(L+2))];
else
      L=(M-1)/2; % M is odd
      A1=[1, -1, 0];
      C1=[real(H(1))];
end
k=[1:L]';
% initialize B and A arrays
B=zeros(L, 2); A=ones(L, 3);
```

```
% compute denominator coefficients
A(1:L, 2)=-2*cos(2*pi*k/M); A=[A; A1];
% compute numerator coefficients
B(1:L, 1)=cos(phah(2:L+1));
B(1:L, 2)=-cos(phah(2:L+1)-(2*pi*k/M));
% compute gain coefficients
C=|2*magH(2:L+1), C1|';
```

9. 变滤波器直接形式为全零点格形结构

```
dir2latc.m
function [K]=dir2latc(b)
% FIR Direct form to All-Zero Lattice form Conversion
% -------------------------------------------
% [K]=dir2latc(b)
% K=Lattice filter coefficients (reflection coefficients)
% b=FIR direct form coefficients (impulse response)
%
M=length(b);
K=zeros(1, M);
b1=b(1);
if b1==0
        error('b(1) is equal to zero')
end
K(1)=b1; A=b/b1;
for m=M:-1:2
        K(m)=A(m);
        J=fliplr(A);
        A=(A-K(m)*J)/1-K(m)*K(m));
        A=A(1:m-1);
end
```

10. 变滤波器全零点格形结构为直接形式

```
latc2dir.m
function [b]=latc2dir(K)
% All-Zero Lattice form to FIR Direct form Conversion
% -------------------------------------------
% [b]=latc2dir(K)
% b=FIR direct form coefficients (impulse response)
```

```
% K=Lattice filter coefficients (reflection coefficients)
%
M=length(b);
J=1; A=1;
for m=2:M
        A=[A, 0]+conv([0, K(m)], J);
        J=fliplr(A);
end
b=A * K(1);
```

11. FIR 滤波器的格形结构实现

```
latcfilt.m
function [y]=latcfilt(K, x)
% LATTICE form realization of FIR filters
% -------------------------
% y=latcfilt(K, x)
% y=output sequence
% K=LATTICE filter (reflection) coefficients array
% x=input sequence
%
Nx=length(x)-1;
x=K(1) * x;
M=length(K)-1; K=K(2:M+1);
fg=[x; [0 x(1:Nx)]];
for m=1:M
        fg=[1, K(m); K(m), 1] * fg;
        fg(2, :)=[0 fg(2, 1:Nx)];
end
y=fg(1, :);
```

3.5.1 IIR 滤波器结构

例 3.30 有一滤波器

$$16y(n)+12y(n-1)+2y(n-2)-4y(n-3)-y(n-4)$$
$$=x(n)-3x(n-1)+11x(n-2)-27x(n-3)+18x(n-4)$$

首先将直接形式转换成级联和并联形式，然后求出这 3 种形式表示时的单位冲激响应。

解 这里用到了我们设计的扩展函数 dir2cas、dir2par、casfiltr、parfiltr 等，MATLAB 程序为 ex3030.m：

```
% Example 3.30
%
```

```
b=[1, -3, 11, -27, 18]; a=[16, 12, 2, -4, -1];
[b0, B, A]=dir2cas(b, a);
[C, B1, A1]=dir2par(b, a);
N=24; n=1:N+1;
delta=impseq(0, 0, N);
h1=filter(b, a, delta);
h2=casfiltr(b0, B, A, delta);
h3=parfiltr(C, B1, A1, delta);
figure(1)
subplot(3, 1, 1)
plot(n, h1)
title('Direct form structure')
subplot(3, 1, 2)
plot(n, h2)
title('Cascade structure')
subplot(3, 1, 3)
plot(n, h3)
title('Parallel form structure')
```

执行后可得到如图 3.33 所示的单位冲激响应。从图中可以看出，这 3 种形式的输出响应是一致的。

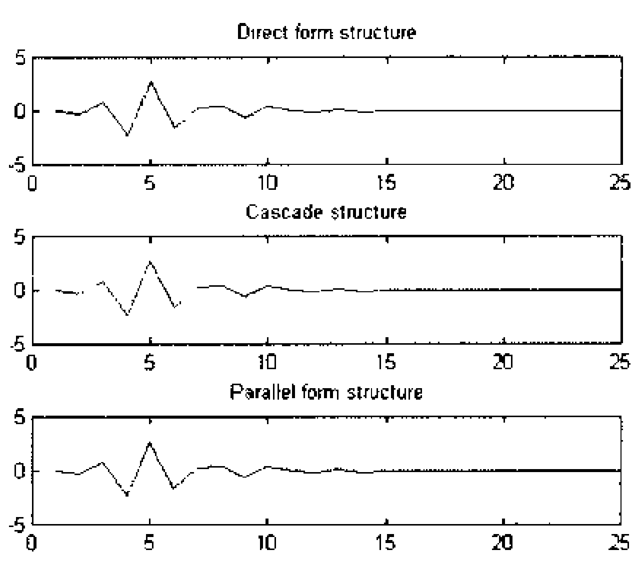


图 3.33 IIR 滤波器 3 种形式的冲激响应

3.5.2 FIR 滤波器结构

例 3.31 FIR 滤波器传递函数为

$$H(z) = 1 + 16\frac{1}{16}z^{-4} + z^{-8}$$

确定并画出滤波器的直接、线性相位和级联形式结构。

解 (1) 直接形式：差分方程为

$$y(n) = x(n) + 16.0625x(n-4) + x(n-8)$$

因此直接形式结构如图 3.34(a)所示。

(2) 线性相位形式：上述差分方程可改写成

$$y(n) = [x(n) + x(n-8)] + 16.0625x(n-4)$$

因此得到如图 3.34(b)所示的线性相位形式。

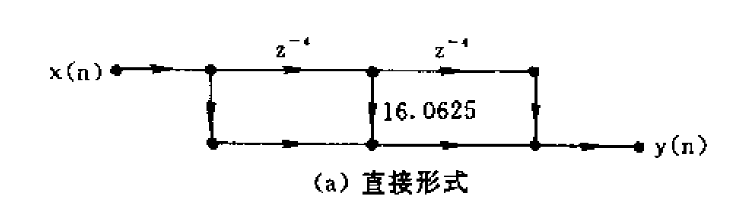
(3) 级联形式：其系数阵可由 MATLAB 计算，MATLAB 程序为 ex3031.m：

```
% Example 3.31
%
b=[1, 0, 0, 0, 16+1/16, 0,0, 0, 1];
[b0, B, A]=dir2cas(b, 1)
```

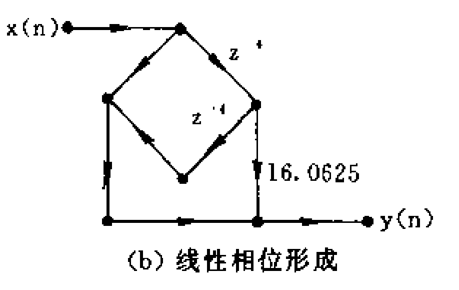
执行后得

```
b0=
  1
B=
  1.0000     2.8284    4.000
  1.0000     0.7071    0.2500
  1.0000    -0.7071    0.2500
  1.0000    -2.8284    4.000
A=
  1    0    0
  1    0    0
  1    0    0
  1    0    0
```

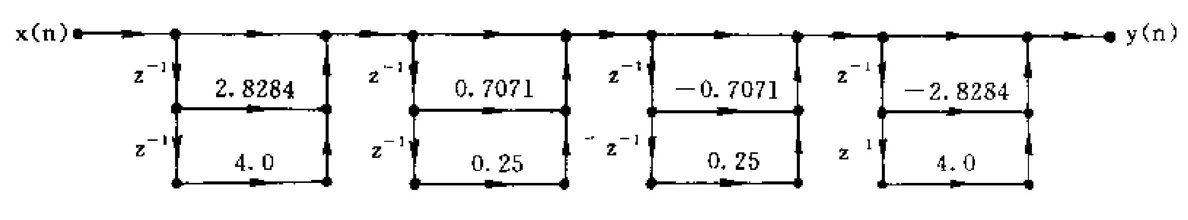
因此得到级联形式，如图 3.34(c) 所示。



(a) 直接形式



(b) 线性相位形成



(c) 级联形式

图 3.34 FIR 滤波器结构

例 3.32 令 $h(n)=\{1, 2, 3, 2, 1\}/9$，确定并画出 FIR 的频率取样形式。

解 首先由 MATLAB 的 dir2fs 函数，将 h(n)直接形式转换成频率取样形式。MAT-

LAB 程序为 ex3032.m：

```
% Example 3.32
%
h=[1, 2, 3, 2, 1]/9;
[C, B, A]=dir2fs(h)
```

执行后得

```
C=
  0.5818
  0.0849
  1.0000
B=
  -0.8090       0.8090
  0.3090     -0.3090
A=
  1.0000     -0.6180     1.0000
  1.0000      1.6180     1.0000
  1.0000     -1.0000     0
```

然后，由于M=5为奇数，因此有

$$H(z)=\frac{1-z^{-5}}{5}\left[0.5818\times\frac{-0.809+0.809z^{-1}}{1-0.618z^{-1}+z^{-2}}+0.0849\times\frac{0.309-0.309z^{-1}}{1+1.618z^{-1}+z^{-2}}+\frac{1}{1-z^{-1}}\right]$$

参照 1.5 节图 1.10 可得到 FIR 的频率取样形式，如图 3.35 所示。

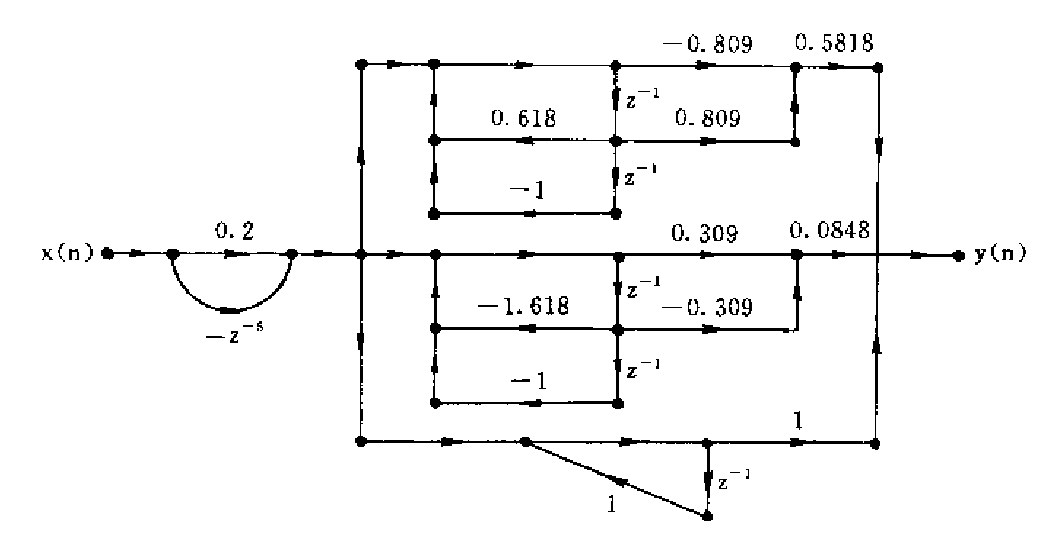


图 3.35 FIR 的频率取样形式(M=5)

3.5.3 格形滤波器结构

例 3.33 FIR 滤波器的差分方程为

$$y(n)=2x(n)+\frac{13}{12}x(n-1)+\frac{5}{4}x(n-2)+\frac{2}{3}x(n-3)$$

确定其格形结构，并画出直接形式和格形形式的冲激响应。

解 MATLAB 程序为 ex3033.m：

```
% Example 3.33
%
b=[2, 13/12, 5/4, 2/3];
K=dir2latc(b)
[x, n]=impseq(0, 0, 30);
y1=filter(b, 1, x);
y2=latcfilt(K, x);
figure(1)
subplot(2, 1, 1)
plot(n, y1)
title('Direct form')
subplot(2, 1, 2)
plot(n, y2)
title('Lattice form')
```

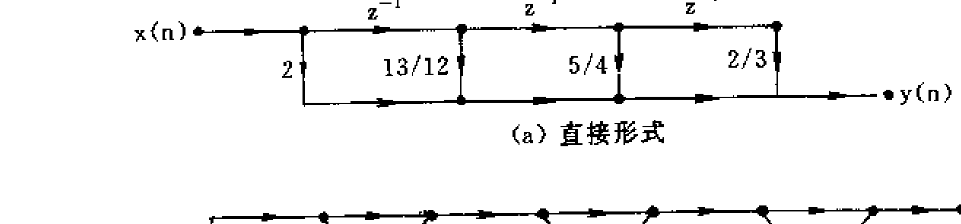
执行后得

```
K=
   2.0000    0.2500    0.5000    0.3333
```

因此格形滤波器系数

$$k_0=2\quad k_1=\frac{1}{4}\quad k_2=\frac{1}{2}\quad k_3=\frac{1}{3}$$

格形滤波器如图 3.36 所示。



(a) 直接形式

(b) 格形形式

图 3.36 FIR 滤波器结构

执行 ex3032还得到 FIR 的冲激响应，如图 3.37 所示，从图中可以看出，这两种形式的冲激响应是一致的。

例 3.34 考虑全极点 IIR 滤波器

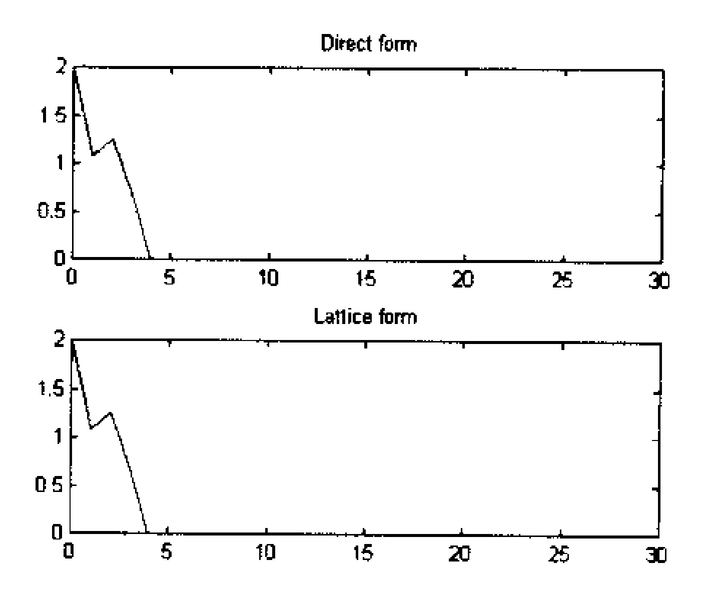


图 3.37 FIR 滤波器的冲激响应

$$H(z)=\frac{1}{1+\frac{13}{24}z^{-1}+\frac{5}{8}z^{-2}+\frac{1}{3}z^{-3}}$$

确定其格形滤波器结构。

解 MATLAB 程序为 ex3034.m：

```
% Example 3.34
%
a=[1, 13/24, 5/8, 1/3];
K=dir2latc(a)
```

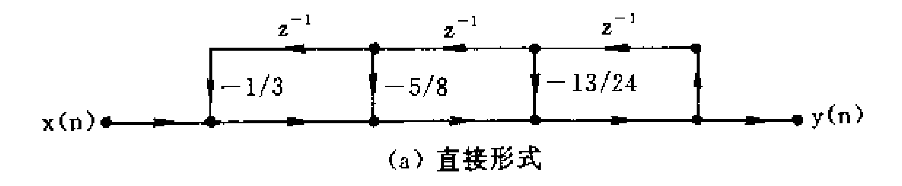
执行后得

```
K=
   1.0000    0.2500    0.5000  0.3333
```

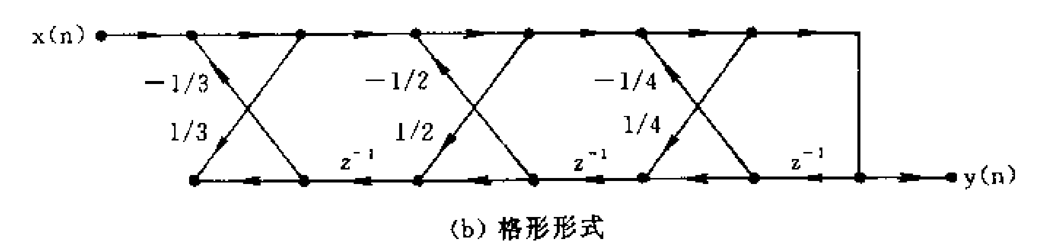
因此

$$k_1=\frac{1}{4}\quad k_2=\frac{1}{2}\quad k_3=\frac{1}{3}$$

k_0默认为 1。其格形结构如图 3.38所示。



(a) 直接形式



(b) 格形形式

图 3.38 FIR 滤波器结构

3.6 FIR 滤波器设计

MATLAB 的扩展函数有 6 种。

1. 计算滤波器振幅响应——线性相位 FIR 滤波器类型1

```
hr_type1.m
function [Hr, w, a, L]=hr_type1(h);
% Computes Amplitude response of Type -1 LP FIR filter
% ---------------------------
% [Hr, w, a, L]=he_type1(h)
% Hr=Amplitude Response
% w=500 frequencies between [0 pi] over which Hr is computed
% a=Type -1 LP filter coefficients
% L=Order of Hr
% h=Type -1 LP filter impulse response
%
M=length(h);
L=(M-1)/2;
a=[h(L+1) 2*h(L:-1:1)];
n=[0:L];
w=[0:500]'*pi/500;
Hr=cos(w*n)*a';
```

2. 计算滤波器振幅响应——线性相位 FIR 滤波器类型2

```
hr_type2.m
function [Hr, w, b, L]=hr_type2(h);
% Computes Amplitude response Hr(w) of a Type -2 LP FIR filter
% ---------------------------
% [Hr, w, b, L]=hr_type2(h)
% Hr=Amplitude Response
% w=frequencies between [0 pi] over which Hr is computed
% b=Type -2 LP filter coefficients
% L=Order of Hr
% h=Type -2 LP filter impulse response
%
M=length(h);
```

```
L=M/2;
b=2*[h(L:-1:1)];
n=[1:L]; n=n-0.5;
w=[0:500]'*pi/500;
Hr=cos(w*n)*b';
```

3. 计算滤波器振幅响应——线性相位 FIR 滤波器类型3

```
hr_type3.m
function [Hr, w, c, L]=hr_type3(h);
% Computes Amplitude response of Type -3 LP FIR filter
% ---------------------------
% [Hr, w, c, L]=hr_type3(h)
% Hr=Amplitude Response
% w=frequencies between [0 pi] over which Hr is computed
% c=Type -3 LP filter coefficients
% L=Order of Hr
% h=Type -3 LP filter impulse response
%
M=length(h);
L=(M-1)/2;
c=[2*h(L+1:-1:1)];
n=[0:L];
w=[0:500]'*pi/500;
Hr=sin(w*n)*c';
```

4. 计算滤波器振幅响应——线性相位 FIR 滤波器类型4

```
hr_type4.m
function [Hr, w, d, L]=hr_type4(h);
% Computes Amplitude response Hr(w) of a Type -4 LP FIR filter
% ---------------------------
% [Hr, w, d, L]=he_type4(h)
% Hr=Amplitude Response
% w=500 frequencies between [0 pi] over which Hr is computed
% d=Type -4 LP filter coefficients
% L=Order of Hr
% h=Type -4 LP filter impulse response
%
M=length(h);
```

```
L=M/2;
d=2*[h(L:-1:1)];
n=[1:L]; n=n-0.5;
w=[0:500]'*pi/500;
Hr=sin(w*n)*d';
```

5. 产生理想低通滤波器的冲激响应

```
ideal_lp.m
function hd=ideal_lp(wc, M);
% Ideal Lowpass filter computation
% -----------------------------
% [hd]=ideal_lp(wc, M)
% hd=ideal impulse response betwwn 0 to M-1
% wc=cutoff frequency in redians
% M=length of the ideal filter
%
alpha=(M-1)/2;
n=[0:(M-1)];
m=n-alpha+eps;
hd=sin(wc*m)./(pi*m);
```

6. freqz 的修正

freqz_m 函数可获得滤波器的幅值响应(绝对和相对)、相位响应及群迟延响应。

```
freqz_m.m
function [db, mag, pha, grd, w]=freqz_m(b, a);
% Modified version of freqz subroutine
% -----------------------------
% [db, mag, pha, grd, w]=freqz m(b, a);
% db=Relative magnitude in dB computed over 0 to pi radians
% mag=absolute magnitude computed over 0 to pi radians
% pha=Phase response in radians over 0 to pi radians
% grd=Group delay over 0 to pi radians
% w=501 frequency samples between 0 and pi redians
% b=numerator polynomial of H(z)   (for FIR: b=h)
% a=denominator polynaomial of H(z)     (for FIR: a=[1])
%
[H, w]=freqz(b, a, 1000, 'whole');
H=(H(1:501))'; w=(w(1:501))';
```

```
mag=abs(H);
db=20 * log10((mag+eps)/max(mag));
pha=angle(H);
grd=grpdelay(b, a, w);
```

3.6.1 线性相位 FIR 滤波器特性

例 3.35 令 h(n)={−4, 1, −1, −2, 5, 6, 5, −2, −1, 1, −4}，确定滤波器的振幅响应 $H_r(\omega)$。

解 由于 M=11，且

$$h(n) = h(M-1-n)$$

α=(11−1)/2=5，因此这是线性相位 FIR 滤波器(类型 1)，故可采用 hr_type1 函数求得振幅响应。MATLAB 程序为 ex3035. m：

```
% Example 3.35
%
h=[-4, 1, -1, -2, 5, 6, 5, -2, -1, 1, -4];
M=length(h); n=0:M-1;
[Hr, w, a, L]=hr_type1(h);
figure(1)
subplot(2, 1, 1); stem(n, h);
title('Impulse Response')
subplot(2, 1, 2); plot(w/pi, Hr); grid
title('Type-1 Amplitude Response')
```

执行后得到如图 3.39 所示的振幅响应曲线。

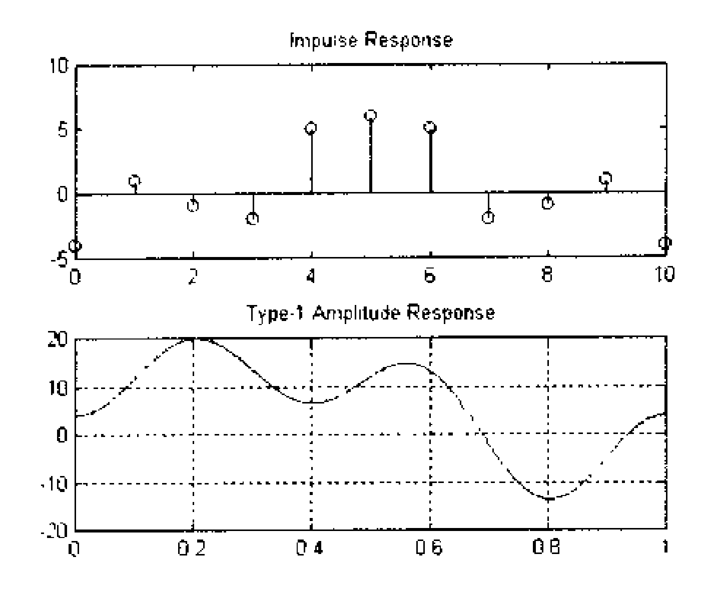


图 3.39 线性相位 FIR 滤波器(类型 1)振幅响应

例 3.36 令 h(n)={−4, 1, −1, −2, 5, 6, 6, 5, −2, −1, 1, −4}，确定滤波器的

振幅响应 $H_r(\omega)$。

解　由于 M=12，α=(12−1)/2=5.5，h(n)对称，因此这是线性相位 FIR 滤波器(类型2)，可利用 hr_type2 函数求得振幅响应 $H_r(\omega)$。MATLAB 程序为 ex3036.m：

```
% Example 3.36
%
figure(1)
h=[-4, 1, -1, -2, 5, 6, 6, 5, -2, -1, 1, -4];
M=length(h); n=0:M-1;
[Hr, w, b, L]=hr_type2(h);
subplot(2, 1, 1); stem(n, h);
title('Impulse Response')
subplot(2, 1, 2); plot(w/pi, Hr); grid
title('Type-2 Amplitude Response')
```

执行后得到如图 3.40 所示的振幅响应曲线。

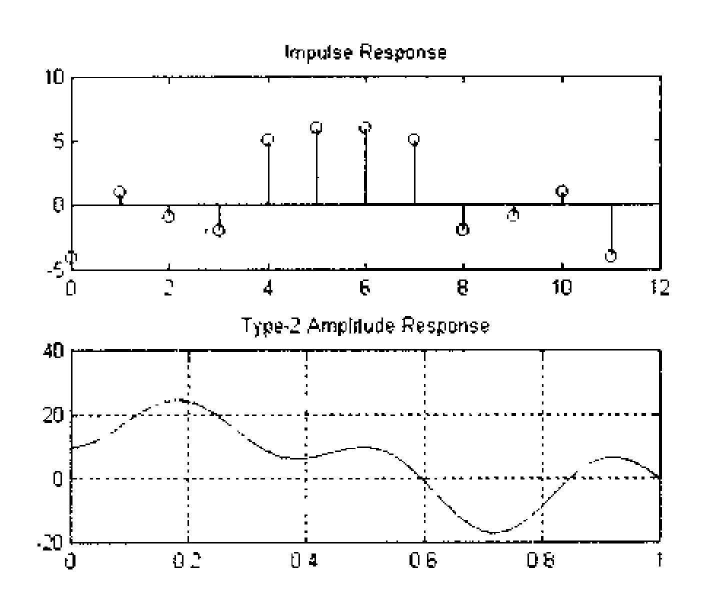


图 3.40　线性相位 FIR 滤波器(类型 2)振幅响应

例 3.37　令 h(n)={−4, 1, −1, −2, 5, 0, −5, 2, 1, −1, 4}，确定滤波器的振幅响应 $H_r(\omega)$。

解　由于 M=11，α=(11−1)/2=5，且 h(n)具有反对称性，因此这是一个线性相位 FIR 滤波器(类型3)，可利用 hr_type3 函数求解振幅响应。MATLAB 程序为 ex3037.m：

```
% Example 3.37
%
figure(1)
h=[-4, 1, -1, -2, 5, 0, -5, 2, 1, -1, 4];
M=length(h); n=0:M-1;
[Hr, w, c, L]=hr_type3(h);
```

```
subplot(2, 1, 1); stem(n, h);
title('Impulse Response')
subplot(2, 1, 2); plot(w/pi, Hr); grid
title('Type-3 Amplitude Response')
```

执行后得如图 3.41 所示的振幅响应曲线。

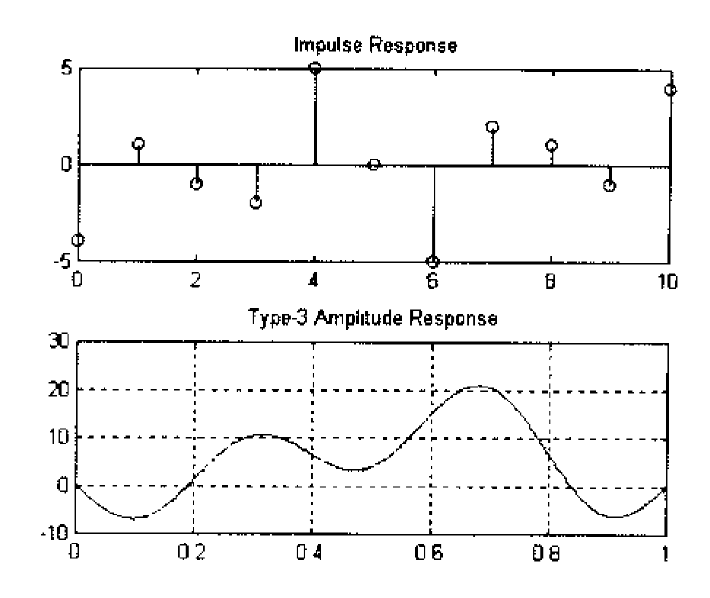


图 3.41　线性相位 FIR 滤波器(类型 3)振幅响应

例 3.38　令 $H(n)=\{-4, 1, -1, -2, 5, 6, -6, -5, 2, 1, -1, 4\}$，确定滤波器的振幅响应 $H_r(\omega)$。

解　由于 $M=12$，$\alpha=(12-1)/2=5.5$，且 $h(n)$ 为反对称特性，因此这是一个线性相位 FIR 滤波器(类型4)，可利用 hr_type4 函数求解振幅响应。MATLAB 程序为ex3038.m：

```
% Example 3.38
%
figure(1)
h=[-4, 1, -1, -2, 5, 6, -6, -5, 2, 1, -1, 4];
M=length(h); n=0:M-1;
[Hr, w, d, L]=hr_type4(h);
subplot(2, 1, 1); stem(n, h);
title('Impulse Response')
subplot(2, 1, 2); plot(w/pi, Hr); grid
title('Type-4 Amplitude Response')
```

执行后得如图 3.42 所示的振幅响应曲线。

3.6.2　利用窗函数设计 FIR 滤波器

例 3.39　设计具有指标

$$\omega_p = 2.5\pi \qquad R_p = 0.25\ \text{dB}$$

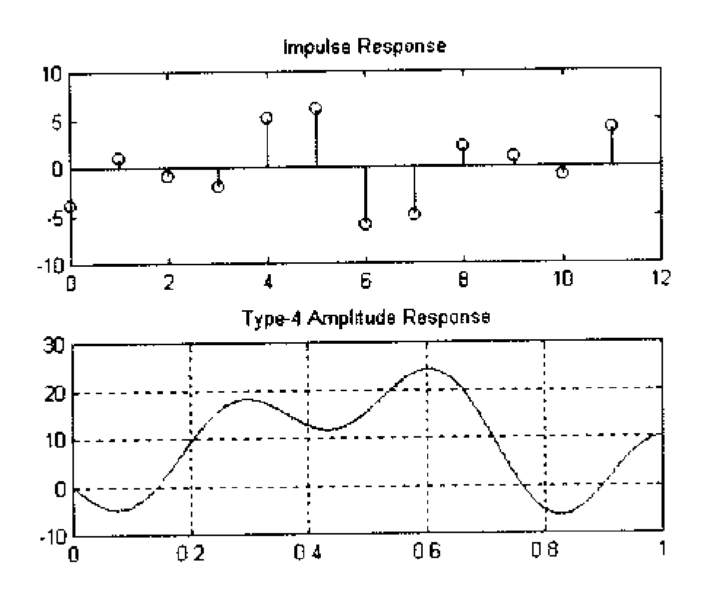


图 3.42 线性相位 FIR 滤波器(类型 4)振幅响应

$$\omega_s = 0.3\pi \qquad A_s = 50 \text{ dB}$$

的低通数字 FIR 滤波器。选择合适的窗函数，确定冲激响应，并画出滤波器的频率响应。

解 从表 1.2 可以看出，对 $A_s=50$ dB，可采用 Hamming 和 Blackman 窗。这里选择 Hamming 窗。

在设计过程中，并没有用到 $R_p=0.25$ dB 值，因此设计后必须对此进行校验。

MATLAB 程序为 ex3039.m：

```
% Example 3.39
%
wp=0.2 * pi; ws=0.3 * pi;
tr_width=ws - wp
M=ceil(6.6 * pi/tr_width) + 1
n=[0:1:M-1];
wc=(ws+wp)/2
hd=ideal_lp(wc, M);
w_ham=(hamming(M))';
h=hd .* w_ham;
[db, mag, pha, grd, w]=freqz_m(h, [1]);
delta_w=2 * pi/1000;
Rp=-(min(db(1:1:wp/delta_w+1)))            % Passband Ripple
As=-round(max(db(ws/delta_w+1:1:501)))  % Min Stopband attenuation

% plots
figure(1)
subplot(2, 2, 1); stem(n, hd); title('Ideal Impulse Response')
```

```
axis([0 M-1 -0.1 0.3]); ylabel('hd(n)')
subplot(2, 2, 2); stem(n, w_ham); title('Hamming Window')
axis([0 M-1 0 1.1]); ylabel('w(n)')
subplot(2, 2, 3); stem(n, h); title('Actual Impulse Response')
axis([0 M-1 -0.1 0.3]); ylabel('h(n)')
subplot(2, 2, 4); plot(w/pi, db); title('Magnitude Response in dB'); grid
axis([0 1 -100 10]); ylabel('Decibels')
set(gca, 'XTickMode', 'manual', 'XTick', [0, 0.2, 0.3, 1])
set(gca, 'YTickMode', 'manual', 'YTick', [-50, 0])
set(gca, 'YTickLabelMode', 'manual', 'YTickLabels', ['50'; ' 0'])
```

执行后得到一些设计参数

```
tr_width=
  0.3142
M=
  67
wc=
  0.7854
Rp=
  0.0394
As=
  52
```

同时产生如图 3.43 所示的结果。其中左上图为理想低通滤波器的冲激响应，右上图为 Hamming 窗，左下图为加窗后的滤波器冲激响应，右下图为滤波器的幅频特性。

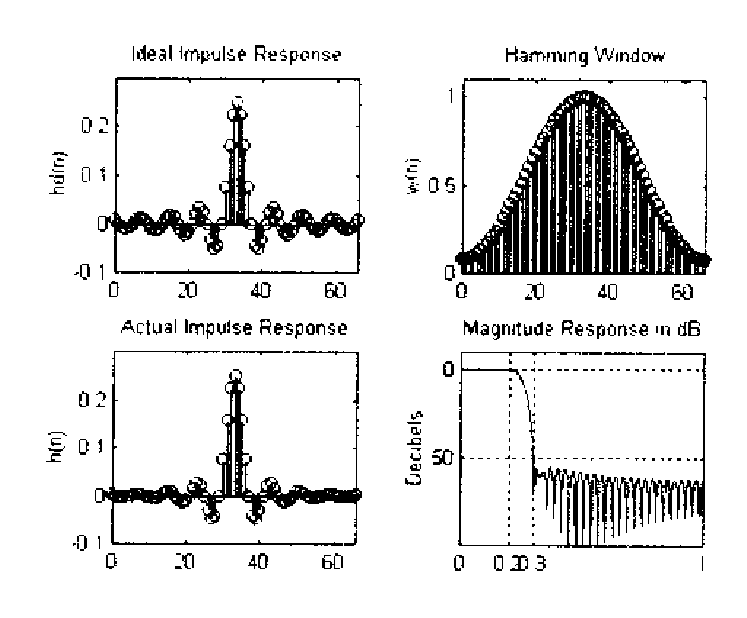


图 3.43　低通滤波器设计

例 3.40　设计数字带通滤波器，指标为

低阻带：$\omega_{1s}=0.2\pi$，$A_s=60$ dB

低通带：$\omega_{1p}=0.35\pi$，$R_p=1$ dB

高通带：$\omega_{2p}=0.65\pi$，$R_p=1$ dB

高阻带：$\omega_{2s}=0.8\pi$，$A_s=60$ dB

解 设选用 Blackman 窗，$\Delta\omega_1 \triangleq \omega_{1p}-\omega_{1s}$，$\Delta\omega_2 \triangleq \omega_{2s}-\omega_{2p}$，则 $\Delta\omega_1=\Delta\omega_2$。MATLAB 程序为 ex3040.m：

```
% Example 3.40
%
ws1=0.2*pi; wp1=0.35*pi;
wp2=0.65*pi; ws2=0.8*pi;
As=60;
tr_width=min((wp1-ws1), (ws2-wp2))
M=ceil(11*pi/tr_width) + 1
n=[0:1:M-1];
wc1=(ws1+wp1)/2; wc2=(wp2+ws2)/2;
hd=ideal_lp(wc2, M) - ideal_lp(wc1, M);
w_bla=(blackman(M))';
h=hd .* w_bla;
[db, mag, pha, grd, w]=freqz_m(h, [1]);
delta_w=2*pi/1000;
Rp=-min(db(wp1/delta_w+1:1:wp2/delta_w))
As=-round(max(db(ws2/delta_w+1:1:501)))

% plots
figure(1);
subplot(2, 2, 1); stem(n, hd); title('Ideal Impulse Response')
axis([0 M-1 -0.4 0.5]); ylabel('hd(n)')
subplot(2, 2, 2); stem(n, w_bla); title('Blackman Window')
axis([0 M-1 0 1.1]); ylabel('w(n)')
subplot(2, 2, 3); stem(n, h); title('Actual Impulse Response')
axis([0 M-1 -0.4 0.5]); ylabel('h(n)')
subplot(2, 2, 4); plot(w/pi, db);
title('Magnitude Response in dB'); grid;
ylabel('Decibels')
axis([0 1 -150 10]);
set(gca, 'XTickMode', 'manual', 'XTick', [0, 0.2, 0.35, 0.65, 0.8, 1])
set(gca, 'YTickMode', 'manual', 'YTick', [-60, 0])
set(gca, 'YTickLabelMode', 'manual', 'YTickLabels', ['60'; ' 0'])
```

执行后得到一些参数

```
tr_width=
  0.4712
M=
  75
Rp=
  0.0030
As=
  75
```

同时得到如图 3.44 所示的滤波器特性。

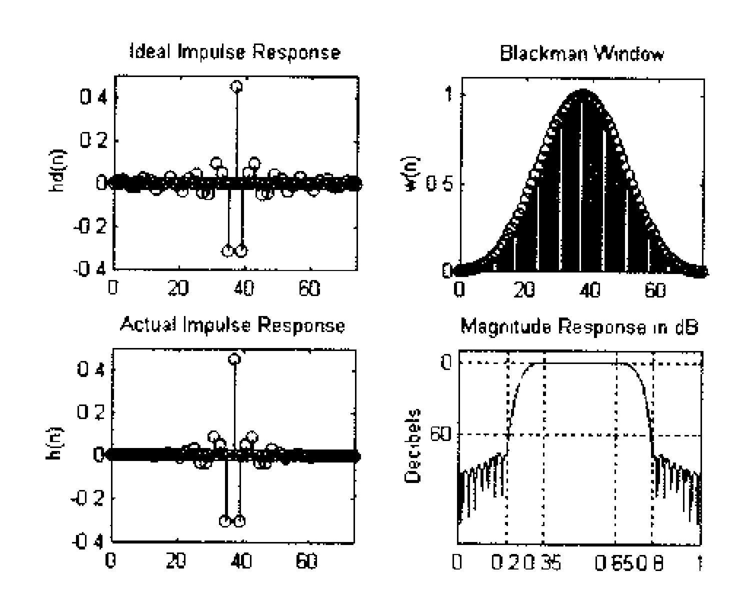


图 3.44 通带滤波器设计

例 3.41 设理想带阻滤波器频率响应为

$$H_e(e^{j\omega}) = \begin{cases} 1 & 0 \leqslant \omega < \pi/3 \\ 0 & \pi/3 \leqslant |\omega| \leqslant 2\pi/3 \\ 1 & 2\pi/3 < |\omega| \leqslant \pi \end{cases}$$

利用 Kaiser 窗函数，设计长度为 45 的带阻滤波器，使阻带衰减为 60 dB。

解 参数β可由下式确定

$$\beta = 0.1102 \times (As - 8.7)$$

因此实现滤波器设计的 MATLAB 程序为 ex3041.m：

```
% Example 3.41
%
M=45; As=60; n=[0:1:M-1];
beta=0.1102 * (As-8.7)
w_kai=(kaiser(M, beta))';
wc1=pi/3; wc2=2 * pi/3;
hd=ideal_lp(wc1, M) + ideal_lp(pi, M) - ideal_lp(wc2, M);
```

```
h=hd . * w_kai;
[db, mag, pha, grd, w]=freqz_m(h, [1]);
%
% plots
%
figure(1);
subplot(2, 2, 1); stem(n, hd); title('Ideal Impulse Response')
axis([-1 M -0.2 0.8]); xlabel('n'); ylabel('hd(n)')
subplot(2, 2, 2); stem(n, w_kai); title('Kaiser Window')
axis([-1 M 0 1.1]); xlabel('n'); ylabel('w(n)')
subplot(2, 2, 3); stem(n, h); title('Actual Impulse Response')
axis([-1 M -0.2 0.8]); xlabel('n'); ylabel('h(n)')
subplot(2, 2, 4); plot(w/pi, db);
title('Magnitude Response in dB'); grid;
xlabel('frequency in pi units'); ylabel('Decibels')
axis([0 1 -80 10]);
set(gca, 'XTickMode', 'manual', 'XTick', [0; 0.333; 0.667; 1])
set(gca, 'XTickLabelMode', 'manual', ...
'XTickLabels', [' 0 '; '1/3'; '2/3'; ' 1 '])
set(gca, 'YTickMode', 'manual', 'YTick', [-60, 0])
set(gca, 'YTickLabelMode', 'manual', 'YTickLabels', ['60'; ' 0'])
```

执行后得

```
bata=
  5.6533
```

同时产生如图 3.45 所示的带阻滤波器特性。从图中不难看出阻带衰减小于 60 dB，为此应

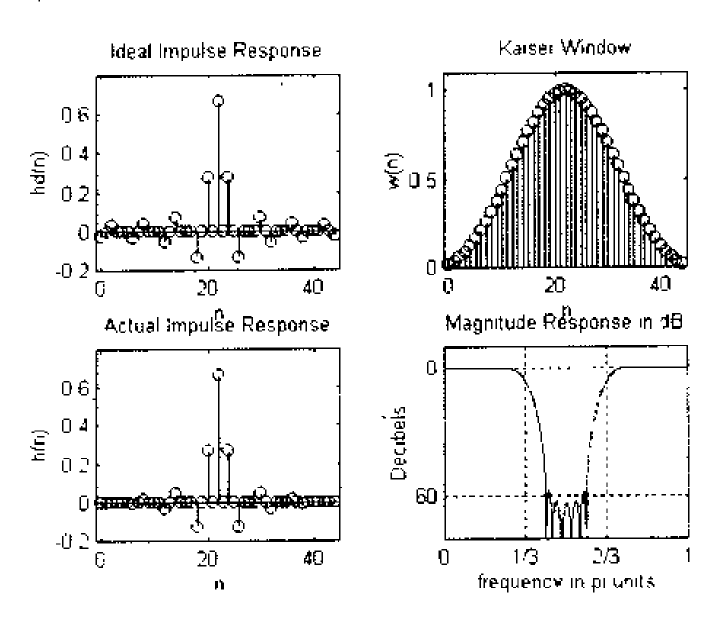


图 3.45　带阻滤波器设计(β=5.6533)

适当增加β，以满足设计要求。当β取 5.9533 时，得到如图 3.46 所示的滤波器特性，这时已满足设计要求。

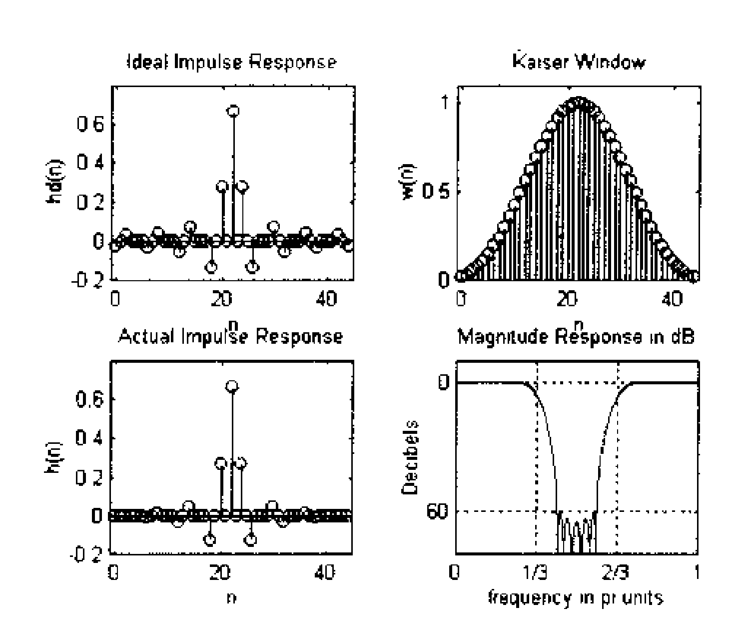


图 3.46 带阻滤波器设计(β=5.9533)

例 3.42 利用 Hanning 窗，设计长度为 25 的数字 Hilbert 变换器。

解 线性相位的 Hilbert 变换器的理想频率响应为

$$H_d(e^{j\omega}) = \begin{cases} -je^{-j\alpha\omega} & 0 < \omega < \pi \\ +je^{-j\alpha\omega} & -\pi < \omega < 0 \end{cases}$$

其冲激响应为

$$h_d(\omega) = \begin{cases} \dfrac{2}{\pi}\dfrac{\sin^2\pi(n-\alpha)/2}{n-\alpha} & n \neq \alpha \\ 0 & n = \alpha \end{cases}$$

由于 M=25，因此所设计滤波器具有类型3。在 MATLAB 中可很容易实现，MATLAB 程序为 ex3042.m：

```
% Example 3.42
%
M=25; alpha=(M-1)/2;
n=0:M-1;
hd=(2/pi)*((sin((pi/2)*(n-alpha)).^2)./(n-alpha));
hd(alpha+1)=0;
w_han=(hanning(M))';
h=hd.*w_han;
[Hr, w, P, L]=Hr_Type3(h);
%
% plots
%
```

```
figure(1);
subplot(2, 2, 1); stem(n, hd); title('Ideal Impulse Response')
axis([-1 M -1.2 1.2]); ylabel('hd(n)')
subplot(2, 2, 2); stem(n, w_han); title('Hanning Window')
axis([-1 M 0 1.2]); ylabel('w(n)')
subplot(2, 2, 3); stem(n, h); title('Actual Impulse Response')
axis([-1 M -1.2 1.2]); xlabel('n'); ylabel('h(n)')
w=w'; Hr=Hr';
w=[-fliplr(w), w(2:501)]; Hr=[-fliplr(Hr), Hr(2:501)];
subplot(2, 2, 4); plot(w/pi, Hr); title('Amplitude Response'); grid;
xlabel('frequency in pi units'); ylabel('Hr')
axis([-1 1 -1.1 1.1]);
set(gca, 'XTickMode', 'manual', 'XTick', [-1, 0, 1])
set(gca, 'YTickMode', 'manual', 'YTick', [-1, 0, 1])
```

执行后可得如图 3.47 所示的 Hilbert 变换器。

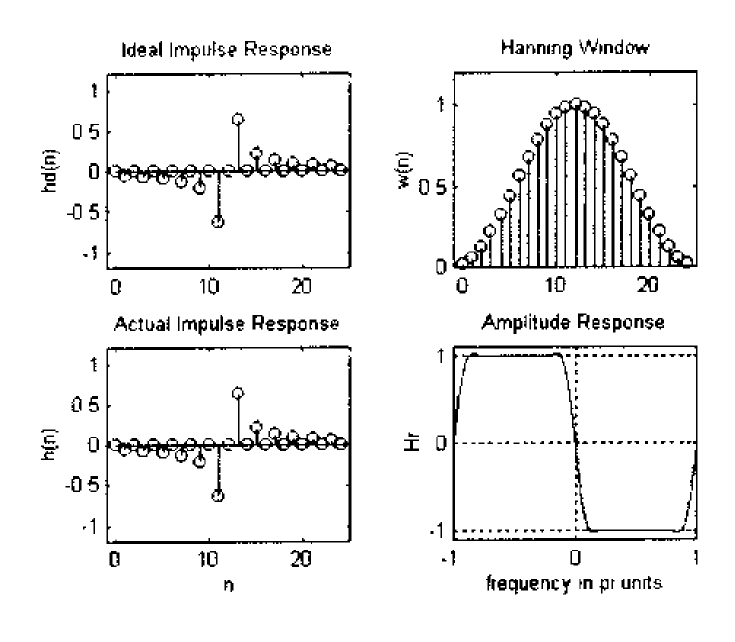


图 3.47 Hilbert 变换器设计

3.6.3 频率取样设计技术

例 3.43 设计低通滤波器

$$\omega_p = 0.2\pi \quad R_p = 0.25\ \text{dB}$$
$$\omega_s = 0.3\pi \quad A_s = 50\ \text{dB}$$

解 选择 M=60，并在过渡带插入两个取样点 $T_1=0.5925$，$T_2=0.1099$，则

$H_r(\omega)=[\text{ones}(1, 7, T_1, T_2, \text{zeros}(1, 43), T_2, T_1, \text{ones}(1, 6)]$

因此 MATLAB 程序为 ex3043.m：

```
% Example 3.43
```

```
%
%        Freq. Samp. Tech. : Lowpass, Optimum method T1 & T2
%        wp=0.2pi, ws=0.3pi, Rp=0.25dB, As=50dB
%        T1=0.5925, T2=0.1099
%
M=60; alpha=(M-1)/2; l=0:M-1; wl=(2*pi/M)*l;
Hrs=[ones(1, 7), 0.5925, 0.1099, zeros(1, 43), 0.1099, 0.5925,
     ones(1, 6)];
Hdr=[1, 1, 0, 0]; wdl=[0, 0.2, 0.3, 1];
k1=0:floor((M-1)/2); k2=floor((M-1)/2)+1:M-1;
angH=[-alpha*(2*pi)/M*k1, alpha*(2*pi)/M*(M-k2)];
H=Hrs.*exp(j*angH);
h=real(ifft(H, M));
[db, mag, pha, grd, w]=freqz_m(h, 1);
[Hr, ww, a, L]=Hr_Type2(h);
%
% plot
%
figure(1)
subplot(2, 2, 1); plot(wl(1:31)/pi, Hrs(1:31), 'o', wdl, Hdr);
axis([0, 1, -0.1, 1.1]); title('Lowpass: M=60, T1=0.59, T2=0.109')
ylabel('Hr(k)')
set(gca, 'XTickMode', 'manual', 'XTick', [0, 0.2, 0.3, 1])
set(gca, 'YTickMode', 'manual', 'YTick', [0, 0.59, 0.109, 1]); grid
subplot(2, 2, 2); stem(l, h); axis([-1, M, -0.1, 0.3])
title('Impulse Response'); xlabel('n'); ylabel('h(n)');
subplot(2, 2, 3); plot(ww/pi, Hr, wl(1:31)/pi, Hrs(1:31), 'o');
axis([0, 1, -0.1, 1.1]); title('Amplitude Response')
xlabel('frequency in pi units'); ylabel('Hr(w)')
set(gca, 'XTickMode', 'manual', 'XTick', [0, 0.2, 0.3, 1])
set(gca, 'YTickMode', 'manual', 'YTick', [0, 0.59, 0.109, 1]); grid
subplot(2, 2, 4); plot(w/pi, db); axis([0, 1, -100, 10]); grid
title('Magnitude Response'); xlabel('frequency in pi units');
ylabel('Decibels');
set(gca, 'XTickMode', 'Manual', 'XTick', [0; 0.2; 0.3; 1]);
set(gca, 'YTickMode', 'Manual', 'YTick', [-63; 0]);
set(gca, 'YTickLabelMode', 'manual', 'YTickLabels', ['63'; ' 0'])
```
执行后可得如图 3.48 所示的结果。

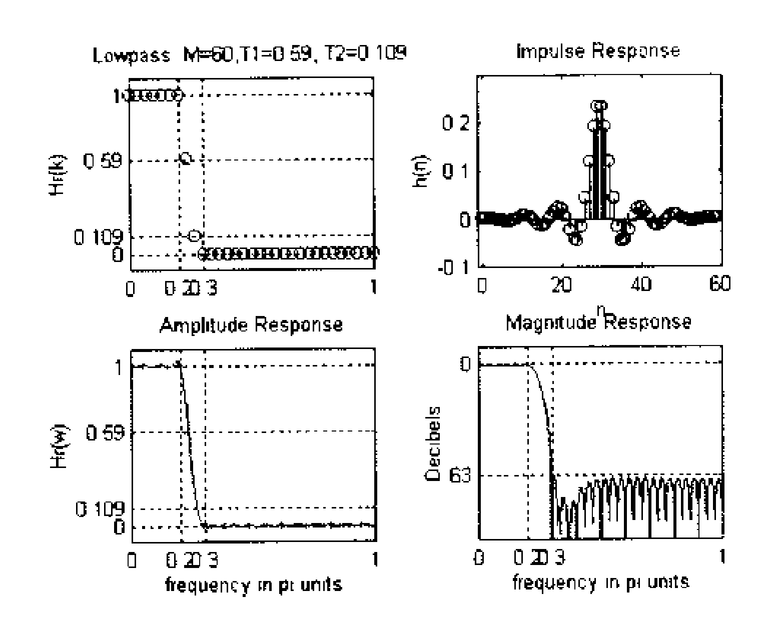


图 3.48 低通滤波器设计

例 3.44 设计带通滤波器

$$\omega_{1s} = 0.2\pi \quad \omega_{2s} = 0.8\pi \quad A_s = 60\ \mathrm{dB}$$
$$\omega_{1p} = 0.35\pi \quad \omega_{2p} = 0.65\pi \quad R_p = 1\ \mathrm{dB}$$

解 选 M=40，过渡带的两个取样点 $T_1=0.1090$，$T_2=0.5941$，则 MATLAB 程序为 ex3044.m：

```
% Example 3.44
%
% Freq. Samp. Tech. : Bandpass, Optimum method T1 & T2
% ws1=0.2pi, wp1=0.35pi, wp2=0.65pi, ws2=0.8pi,
% Rp=1dB, As=60dB
% T2=0.5941, T1=0.1090
%
M=40; alpha=(M-1)/2; l=0:M-1; wl=(2*pi/M)*l;
T1=0.109021; T2=0.59417456;
Hrs=[zeros(1, 5), T1, T2, ones(1, 7), T2, T1, zeros(1, 9), T1, T2,
      ones(1, 7), T2, T1, zeros(1, 4)];
Hdr=[0, 0, 1, 1, 0, 0]; wdl=[0, 0.2, 0.35, 0.65, 0.8, 1];
k1=0:floor((M-1)/2); k2=floor((M-1)/2)+1:M-1;
angH=[-alpha*(2*pi)/M*k1, alpha*(2*pi)/M*(M-k2)];
H=Hrs.*exp(j*angH);
h=real(ifft(H, M));
[db, mag, pha, grd, w]=freqz_m(h, 1);
[Hr, ww, a, L]=Hr_Type2(h);
%
```

```
% plot
%
figure(1)
subplot(2, 2, 1); plot(wl(1:21)/pi, Hrs(1:21), 'o', wdl, Hdr);
axis([0, 1, -0.1, 1.1]); title('Bandpass: M=40, T1=0.5941, T2=0.109')
ylabel('Hr(k)')
set(gca, 'XTickMode', 'manual', 'XTick', [0, 0.2, 0.35, 0.65, 0.8, 1])
set(gca, 'YTickMode', 'manual', 'YTick', [0, 0.59, 0.109, 1]); grid
subplot(2, 2, 2); stem(l, h); axis([-1, M, -0.4, 0.4])
title('Impulse Response'); xlabel('n'); ylabel('h(n)');
subplot(2, 2, 3); plot(ww/pi, Hr, wl(1:21)/pi, Hrs(1:21), 'o');
axis([0, 1, -0.1, 1.1]); title('Amplitude Response')
xlabel('frequency in pi units'); ylabel('Hr(w)')
set(gca, 'XTickMode', 'manual', 'XTick', [0, 0.2, 0.35, 0.65, 0.8, 1])
set(gca, 'YTickMode', 'manual', 'YTick', [0, 0.59, 0.109, 1]); grid
subplot(2, 2, 4); plot(w/pi, db); axis([0, 1, -100, 10]); grid
title('Magnitude Response'); xlabel('frequency in pi units');
ylabel('Decibels');
set(gca, 'XTickMode', 'Manual', 'XTick', [0, 0.2, 0.35, 0.65, 0.8, 1]);
set(gca, 'YTickMode', 'Manual', 'YTick', [-60; 0]);
set(gca, 'YTickLabelMode', 'manual', 'YTickLabels', ['60'; ' 0'])
```

执行后得如图 3.49 所示的结果。

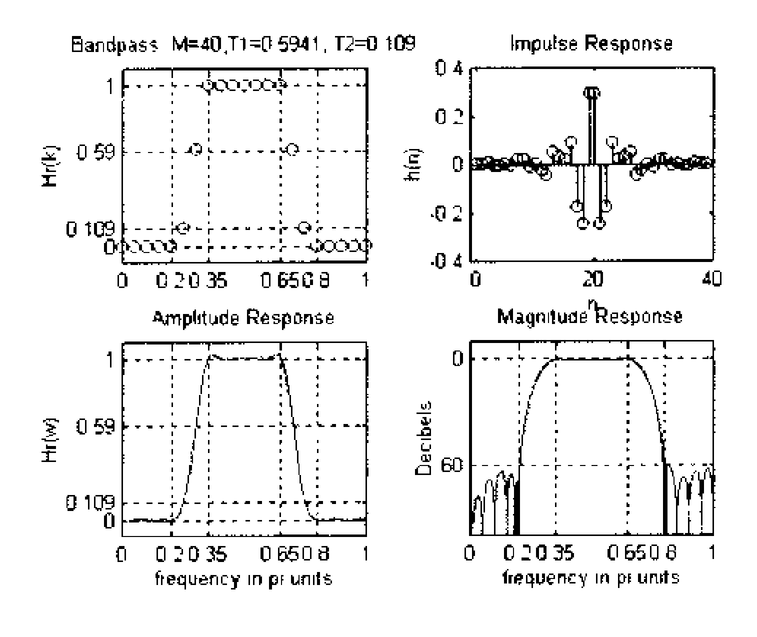


图 3.49 带通滤波器设计

例 3.45 设计高通滤波器

$$\omega_s = 0.6\pi \qquad A_s = 50\ \text{dB}$$

$$\omega_p = 0.8\pi \qquad R_p = 1\ \text{dB}$$

解 选 M=33(必须为奇数)，且在过渡区增加两个取样点 $T_1=0.1095$，$T_2=0.5980$。这时 MATLAB 程序为 ex3045.m：

```
% Example 3.45
%
% Freq. Samp. Tech. ; Highpass, Optimum method T1
% ws=0.6pi, wp=0.8pi, Rp=1dB, As=50dB
% M=33, T1=0.1095; T2=0.5980;
%
M=33; alpha=(M-1)/2; l=0:M-1; wl=(2*pi/M)*l;
T1=0.1095; T2=0.598;
Hrs=[zeros(1,11),T1,T2,ones(1,8),T2,T1,zeros(1,10)];
Hdr=[0,0,1,1]; wdl=[0,0.6,0.8,1];
k1=0:floor((M-1)/2); k2=floor((M-1)/2)+1:M-1;
angH=[-alpha*(2*pi)/M*k1, alpha*(2*pi)/M*(M-k2)];
H=Hrs.*exp(j*angH);
h=real(ifft(H,M));
[db,mag,pha,grd,w]=freqz_m(h,1);
[Hr,ww,a,L]=Hr_Type1(h);
%
% plot
%
figure(1)
subplot(2,2,1); plot(wl(1:17)/pi,Hrs(1:17),'o',wdl,Hdr);
axis([0,1,-0.1,1.1]); title('Highpass: M=33, T1=0.1095, T2=0.598')
ylabel('Hr(k)')
set(gca,'XTickMode','manual','XTick',[0;.6;.8;1])
set(gca,'XTickLabelMode','manual','XTickLabels',[' 0';'.6';'.8';' 1'])
set(gca,'YTickMode','manual','YTick',[0,0.109,0.59,1]); grid
subplot(2,2,2); stem(l,h); axis([-1,M,-0.4,0.4])
title('Impulse Response'); xlabel('n'); ylabel('h(n)');
subplot(2,2,3); plot(ww/pi,Hr,wl(1:17)/pi,Hrs(1:17),'o');
axis([0,1,-0.1,1.1]); title('Amplitude Response')
xlabel('frequency in pi units'); ylabel('Hr(w)')
set(gca,'XTickMode','manual','XTick',[0;.6;.8;1])
set(gca,'XTickLabelMode','manual','XTickLabels',[' 0';'.6';'.8';' 1'])
set(gca,'YTickMode','manual','YTick',[0,0.109,0.59,1]); grid
```

```
subplot(2, 2, 4); plot(w/pi, db); axis([0, 1, -100, 10]); grid
title('Magnitude Response'); xlabel('frequency in pi units');
ylabel('Decibels');
set(gca, 'XTickMode', 'manual', 'XTick', [0; .6; .8; 1])
set(gca, 'XTickLabelMode', 'manual', 'XTickLabels', [' 0'; '.6'; '.8'; ' 1'])
set(gca, 'YTickMode', 'Manual', 'YTick', [-50; 0]);
set(gca, 'YTickLabelMode', 'manual', 'YTickLabels', ['50'; ' 0'])
```

执行后得如图 3.50 所示的结果。

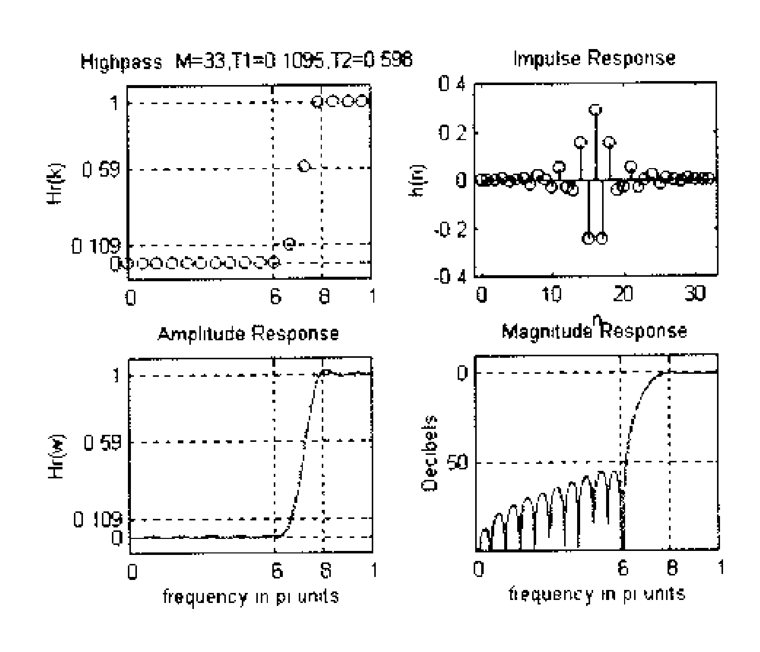


图 3.50 高通滤波器设计

3.6.4 最佳等波纹设计技术

例 3.46 设计低通滤波器

$$\omega_p = 0.2\pi \qquad R_p = 0.25\ \text{dB}$$
$$\omega_s = 0.3\pi \qquad A_s = 50\ \text{dB}$$

画出等波纹滤波器特性。

解 MATLAB 程序为 ex3046.m:

```
% Example 3.46
%
%  Lowpass filter design using PM algorithm
%
wp =0.2*pi; ws =0.3*pi; Rp=0.25; As=50;
delta1=(10^(Rp/20)-1)/(10^(Rp/20)+1);
delta2=(1+delta1)*(10^(-As/20));
deltaH=max(delta1, delta2);
```

```
deltaL=min(delta1, delta2);
weights=[delta2/delta1 1];
deltaf=(ws-wp)/(2*pi);
M=ceil((-20*log10(sqrt(delta1*delta2))-13)/(14.6*deltaf)+1);
M=M+4,           % modification
f=[0 wp/pi ws/pi 1];
m=[1 1 0 0];
h=remez(M-1, f, m, weights);
[db, mag, pha, grd, w]=freqz_m(h, [1]);
delta_w=2*pi/1000; wsi=ws/delta_w+1; wpi=wp/delta_w;
Asd=-max(db(wsi:1:501))
%
% Plots
%
figure(1);
subplot(2, 1, 1); stem([0:1:M-1], h); title('Actual Impulse Response')
axis([0 M-1 -0.1 0.3]); xlabel('n'); ylabel('h(n)')
set(gca, 'XTickMode', 'manual', 'XTick', [0, M-1])
set(gca, 'YTickMode', 'manual', 'YTick', [-0.1:0.1:0.3])
subplot(2, 1, 2); plot(w/pi, db); title('Magnitude Response in dB');
axis([0, 1, -80, 10]);
xlabel('frequency in pi units'); ylabel('DECIBELS')
set(gca, 'XTickMode', 'manual', 'XTick', [0, 0.2, 0.3, 1])
set(gca, 'YTickMode', 'manual', 'YTick', [-50, 0])
set(gca, 'YTickLabelMode', 'manual', 'YTickLabels', ['50'; ' 0']), grid
```

执行后得

```
M=
  47
Asd=
  51.0857
```

并同时产生如图 3.51 所示的等波纹滤波器特性。

例 3.47 设计等波纹带通滤波器

$$\omega_{1s}=0.2\pi \qquad \omega_{1p}=0.35\pi \qquad R_p=1\ \text{dB}$$
$$\omega_{2p}=0.65\pi \qquad \omega_{2s}=0.8\pi \qquad A_s=60\ \text{dB}$$

画出滤波器特性。

解 MATLAB 程序为 ex3047.m:

```
% Example 3.47
%
%   Bandpass filter design using PM algorithm
```

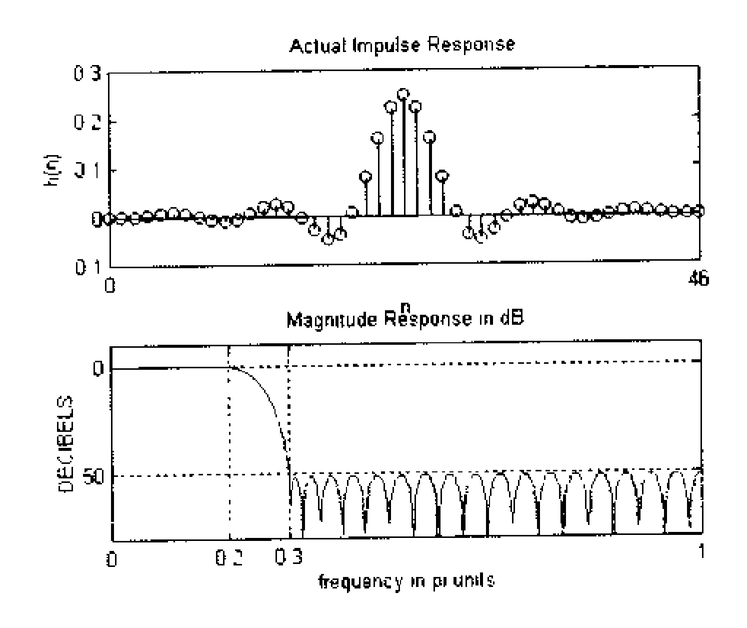


图 3.51 低通等波纹滤波器特性

```
%
ws1=0.2*pi; wp1=0.35*pi; wp2=0.65*pi; ws2=0.8*pi;
Rp=1.0; As=60;
delta1=(10^(Rp/20)-1)/(10^(Rp/20)+1);
delta2=(1+delta1)*(10^(-As/20));
deltaH=max(delta1,delta2); deltaL=min(delta1,delta2);
weights=[1 delta2/delta1 1];
delta_f =min((ws2-wp2)/(2*pi),(wp1-ws1)/(2*pi));
M=ceil((-20*log10(sqrt(delta1*delta2))-13)/(14.6*delta_f)+1);
M=M+1
f=[0 ws1/pi wp1/pi wp2/pi ws2/pi 1];
m=[0 0 1 1 0 0];
h=remez(M-1,f,m,weights);
[db,mag,pha,grd,w]=freqz_m(h,[1]);
delta_w=2*pi/1000;
ws1i=floor(ws1/delta_w)+1; wp1i=floor(wp1/delta_w)+1;
ws2i=floor(ws2/delta_w)+1; wp2i=floor(wp2/delta_w)+1;
Asd=-max(db(1:1:ws1i))
%
% Plots
%
figure(1);
subplot(2,1,1); stem([0:1:M-1],h); title('Actual Impulse Response')
```

```
axis([0, M-1, -0.4, 0.5]); xlabel('n'); ylabel('h(n)')
set(gca, 'XTickMode', 'manual', 'XTick', [0, M-1])
set(gca, 'YTickMode', 'manual', 'YTick', [-0.4:0.2:0.5])
subplot(2, 1, 2); plot(w/pi, db); title('Magnitude Response in dB');
axis([0, 1, -80, 10]); xlabel('frequency in pi units'); ylabel('DECIBELS')
set(gca, 'XTickMode', 'manual', 'XTick', f)
set(gca, 'YTickMode', 'manual', 'YTick', [-60, 0]);
set(gca, 'YTickLabelMode', 'manual', 'YTickLabels', ['60'; ' 0']); grid
```

执行后得

```
M=
  29
Asd=
  61.2843
```

同时得到如图 3.52 所示的滤波器特性。

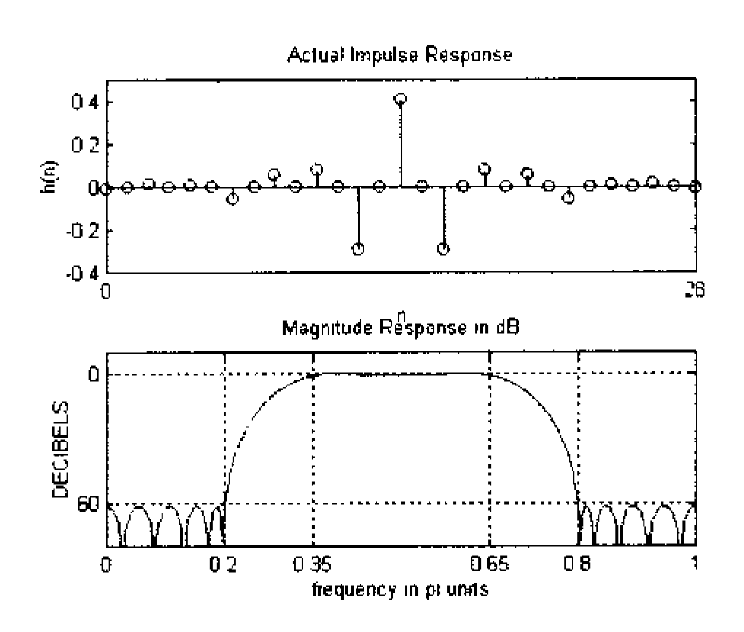


图 3.52 带通等波纹滤波器特性

例 3.48 设计阶梯形滤波器，3 个频带的指标为

频带 1：$0\leqslant\omega\leqslant 0.3\pi$ 增益为 1，$\delta_1=0.01$

频带 2：$0.4\pi\leqslant\omega\leqslant 0.7\pi$ 增益为 0.5，$\delta_2=0.005$

频带 3：$0.8\pi\leqslant\omega\leqslant\pi$ 增益为 0，$\delta_3=0.001$

解 MATLAB 程序为 ex3048.m：

```
% Example 3.48
%
% Staircase filter design using PM algorithm
%
```

```
w1=0; w2=0.3 * pi; delta1=0.01;
w3=0.4 * pi; w4=0.7 * pi; delta2=0.005;
w5=0.8 * pi; w6=pi; delta3=0.001;
deltaH=max([delta1, delta2, delta3]);
deltaL=min([delta1, delta2, delta3]);
weights=[delta3/delta1 delta3/delta2 1];
delta_f=min((w3-w2)/(2 * pi), (w5-w3)/(2 * pi));

% optimum value was found at M=49
M=49;
f=[0 w2/pi w3/pi w4/pi w5/pi 1];
m=[1 1 0.5 0.5 0 0];
h=remez(M-1, f, m, weights);
[db, mag, pha, grd, w]=freqz_m(h, [1]);
delta_w=2 * pi/1000;
w1i=floor(w1/delta_w)+1; w2i=floor(w2/delta_w)+1;
w3i=floor(w3/delta_w)+1; w4i=floor(w4/delta_w)+1;
w5i=floor(w5/delta_w)+1; w6i=floor(w6/delta_w)+1;
Asd=-max(db(w5i:w6i))
%
% Plots
%
figure(1);
subplot(2, 1, 1); stem([0:1:M-1], h); title('Actual Impulse Response')
axis([0, M-1, -0.1, 0.6]); xlabel('n'); ylabel('h(n)')
set(gca, 'XTickMode', 'manual', 'XTick', [0, M-1])
set(gca, 'YTickMode', 'manual', 'YTick', [-0.1:0.1:0.6])
subplot(2, 1, 2); plot(w/pi, db); title('Magnitude Response in dB');
axis([0, 1, -80, 10]); xlabel('frequency in pi units'); ylabel('DECIBELS')
set(gca, 'XTickMode', 'manual', 'XTick', f)
set(gca, 'YTickMode', 'manual', 'YTick', [-60, 0])
set(gca, 'YTickLabelMode', 'manual', 'YTickLabels', ['60'; ' 0']); grid
```

执行后得

```
Asd=
   60.6068
```

同时得如图 3.53 所示的滤波器特性。

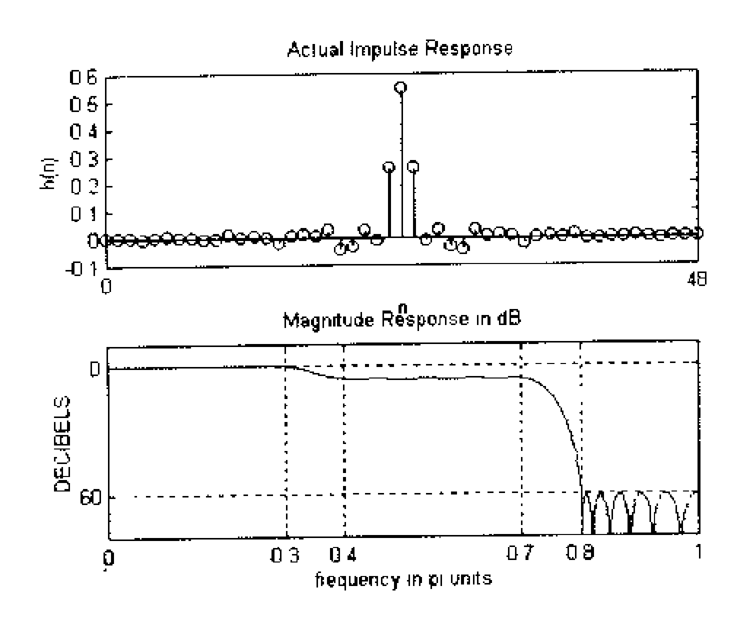


图 3.53　阶梯形等波纹滤波器特性

3.7　IIR 滤波器设计

MATLAB 扩展函数有 13 种。

1. 非归一化 Butterworth 模拟低通滤波器原型设计

```
u_buttap.m
function [b, a]=u_buttap(N, Omegac);
% Unnormalized Butterworth Analog Lowpass Filter Prototype
% ----------------------------
% [b, a]=u_buttap(N, Omegac);
% b=numerator polynomial coefficients of Ha(s)
% a=denominator polynomial coefficients of Ha(s)
% N=Order of the Butterworth Filter
% Omegac=Cutoff frequency in radians/sec
%
[z, p, k]=buttap(N);
p=p * Omegac;
k=k * Omegac^ N;
B=real(poly(z));
b0=k;
b=k * B;
a=real(poly(p));
```

2. 变滤波器直接形式为级联形式

```
sdir2cas.m
function [C, B, A]=sdir2cas(b, a);
% DIRECT - form to CASCADE - form conversin in s - plane
% --------------------------
% [C, B, A]=sdir2cas(b, a)
% C=gain coefficient
% B=K by 3 matrix of real coefficients containing bk's
% A=K by 3 matrix of real coefficients containing ak's
% b=numberator polynomial coefficients of DIRECT form
% a=denominator polynomial coefficients of DIRECT form
%
Na=length(a)-1; Nb=length(b)-1;

% compute gain coefficient C
b0=b(1); b=b/b0;
a0=a(1); a=a/a0;
C=b0/a0;
%
% Denominator second - order sections;
p=cplxpair(roots(a)); K=floor(Na/2);
if K * 2==Na          % Computation when Na is even
    A=zeros(K, 3);
    for n=1:2:Na
        Arow=p(n:1:n+1, :);
        Arow=poly(Arow);
        A(fix((n+1)/2, :)=real(Arow);
    end

elseif Na==1          % Computation when Na=1
     A=[0 real(poly(p))];
else                  % Computation when Na is odd and >1
    A=zeros(K+1, 3);
    for n=1:2:2 * K
        Arow=p(n:1:n+1, :);
        Arow=poly(Arow);
        A(fix(n+1)/2, :)=real(Arow);
    end
```

```
    A(K+1, :)=[0 real(poly(p(Na)))];
end

% Numerator second - order sections:
z=cplxpair(roots(b)); K=floor(Nb/2);
if Nb==0              % Computation when Nb=0
  B=[0 0 poly(z)];
elseif K * 2==Nb      % Computation when Nb is even
    B=zeros(K, 3);
    for n=1:2:Nb
        Brow=z(n:1:n+1, :)
        Brow=poly(Brow);
        B(fix((n+1)/2), :)=real(Brow);
    end

elseif Nb==1          % Computation when Nb=1
     B=[0 real(poly(z))];
else                  % Computation when Nb is odd and >1
    B=zeros(K+1, 3);
    for n=1:2:2 * K
        Brow=z(n:1:n+1, :)
        Brow=poly(Brow);
        B(fix((n+1)/2, :)=real(Brow);
    end
    B(K+1, :)=[0 real(poly(z(Nb)))];
end
```

3. Butterworth 低通滤波器原型设计

```
afd_butt.m
function [b, a]=afd_butt(Wp, Ws, Rp, As);
% Analog Lowpass Filter Design: Butterworth
% -------------------------
% [b, a]=afd_butt(Wp, Ws, Rp, As);
% b=Numberator coefficients of Ha(s)
% a=Denominator coefficients of Ha(s)
% Wp=Passband edge frequency in rad/sec; Wp>0
% Ws=Stopband edge frequency in rad/sec; Ws>Wp>0
% Rp=Passband ripple in +dB; (Rp>0)
% As=Stopband attenuation in +dB; (As>0)
```

```
%
if Wp<=0
        error('Passband edge must be larger than 0')
end
if Ws<=Wp
        error('Stopband edge must be larger than Passband edge')
end
if (Rp<=0)|(As<0)
        error('PB ripple and/or SB attenuation must be larger than 0')
end

N=ceil((log10((10^(Rp/10)-1)/(10^(As/10)-1)))
     /(2*log10(Wp/Ws)));
fprintf('\n * * * Butterworth Filter Order=%2.0f \n', N)
OmegaC=Wp/((10^(Rp/10)-1)^(1/(2*N)));
[b, a]=u_buttap(N, OmegaC);
```

4. freqs 的修正

freqs_m 函数可计算出滤波器的幅值响应(绝对和相对)及相位响应。

```
freqs_m.m
function [db, mag, pha, w]=freqs_m(b, a, wmax);
% Computation of s-domain frequency response: Modified version
% ----------------------------
% [db, mag, pha, w]=freqs_m(b, a, wmax);
% db=Relative magnitude in dh over [0 to wmax]
% mag=Absolute magnitude over [0 to wmax]
% pha=Phase response in radians over [0 to wmax]
% w=array of 500 frequency samples between [0 to wmax]
% b=Numerator polynomial coefficents of Ha(s)
% a=Denominator polynaomial coefficents of Ha(s)
% wmax=Maximum frequency in rad/sec over which response is desired
%
w=[0:500]*wmax/500;
H=freqs(b, a, w);
mag=abs(H);
db=20*log10((mag+eps)/max(mag));
pha=angle(H);
```

5. 非归一化 Chebyshev Ⅰ 型模拟低通滤波器原型设计

```
u_chb1ap.m
function [b, a]=u_chb1ap(N, Rp, Omegac);
% Unnormalized Chebyshev -1 Analog Lowpass Filter Prototype
% -------------------------
% [b, a]=u chb1ap(N, Rp, Omegac);
% b=numberator polynomial coefficients
% a=denominator polynomial coefficients
% N=Order of the Elliptic Filter
% Rp=Passband Ripple in dB; Rp>0
% Omegac=Cutoff frequency in radians/sec
%
[z, p, k]=cheb1ap(N, Rp);
        a=realpoly(p));
        aNn=a(N+1);
        p=p * Omegac;
        a=real(poly(p));
        aNu=a(N+1);
        k=k * aNu/aNn;
        b0=k;
        B=real(poly(z));
        b=k * B;
```

6. Chebyshev Ⅰ 型模拟低通滤波器原型设计

```
afd_chb1.m
function [b, a]=afd_chb1(Wp, Ws, Rp, As);
% Analog Lowpass Filter Design: Chebyshev -1
% -------------------------
% [b, a]=afd_chb1(Wp, Ws, Rp, As);
% h=Numberator coefficients of Ha(s)
% a=Denominator coefficients of Ha(s)
% Wp=Passband edge frequency in rad/sec; Wp>0
% Ws=Stopband edge frequency in rad/sec; Ws>Wp>0
% Rp=Passband ripple in +dB; (Rp>0)
% As=Stopband attenuation in +dB; (As>0)
%
if Wp<=0
```

```
            error('Passband edge must be larger than 0')
end
if Ws<=Wp
            error('Stopband edge must be larger than Passband edge')
end
if (Rp<=0)|(As<0)
            error('PB ripple and/or SB attenuation must be larger than 0')
end

ep=sqrt(10^ (Rp/10)-1);
A=10^ (As/20);
OmegaC=Wp;
OmegaR=Ws/Wp;
g=sqrt(A*A-1)/ep;
N=ceil(log10(g+sqrt(g*g-1))/log10(OmegaR+sqrt(OmegaR
      *OmegaR-1)));
fprintf('\n * * * Chebyshev -1 Filter Order=%2.0f \n', N)
[b, a]=u chb1ap(N, Rp, OmegaC);
```

7. 非归一化 Chebyshev Ⅱ 型模拟低通滤波器原型设计

```
u_chb2ap.m
function [b, a]=u_chb2ap(N, As, Omegac);
% Unnormalized Chebyshev -2 Analog Lowpass Filter Prototype
% ---------------------------
% [b, a]=u_chb2ap(N, As, Omegac);
% b=numerator polynomial coefficients
% a=denominator polynomial coefficients
% N=Order of the Elliptic Filter
% As=Stopband Ripple in dB; As>0
% Omegac=Cutoff frequency in radians/sec
%
[z, p, k]=cheb2ap(N, As);
        a=real(poly(p));
        aNn=a(N+1);
        p=p*Omegac;
        a=real(poly(p));
        aNu=a(N+1);
        b=real(poly(p));
        M=length(b);
```

```
bNn=b(M);
z=z * Omegac;
b=real(poly(z));
bNu=b(M);
k=k * (aNu * bNn)/(aNn * bNu);
b0=k;
b=k * b;
```

8. Chebyshev Ⅱ 型模拟低通滤波器原型设计

```
afd_chb2.m
function [b, a]=afd_chb2(Wp, Ws, Rp, As);
% Analog Lowpass Filter Design: Chebyshev -2
% ---------------------------
% [b, a]=afd_chb2(Wp, Ws, Rp, As);
% b=Numberator coefficients of Ha(s)
% a=Denominator coefficients of Ha(s)
% Wp=Passband edge frequency in rad/sec; Wp>0
% Ws=Stopband edge frequency in rad/sec; Ws>Wp>0
% Rp=Passband ripple in +dB; (Rp>0)
% As=Stopband attenuation in +dB; (As>0)
%
if Wp<=0
        error('Passband edge must be larger than 0')
end
if Ws<=Wp
        error('Stopband edge must be larger than Passband edge')
end
if (Rp<=0)|(As<0)
        error('PB ripple and/or SB attenuation must be larger than 0')
end

ep=sqrt(10^(Rp/10)-1);
A=10^(As/20);
OmegaC=Wp;
OmegaR=Ws/Wp;
g=sqrt(A * A-1)/ep;
N=ceil(log10(g+sqrt(g * g-1))/log10(OmegaR+sqrt(OmegaR
    * OmegaR-1)));
fprintf('\n * * * Chebyshev -2 Filter Order=%2.0f \n', N)
```

```
[b, a]=u_chb2ap(N, As, Ws);
```

9. 非归一化椭圆模拟低通滤波器原型设计

```
u_elipap.m
function [b, a]=u_elipap(N, Rp, As, Omegac);
% Unnormalized Elliptic Analog Lowpass Filter Prototype
% --------------------------
% [b, a]=u_elipap(N, Rp, As, Omegac);
% b=numberator polynomial coefficients
% a=denominator polynomial coefficients
% N=Order of the Elliptic Filter
% Rp=Passband Ripple in dB; Rp>0
% As=Stopband Attebuation in dB; As>0
% Omegac=Cutoff frequency in radians/sec
%
[z, p, k]=ellipap(N, Rp, As);
        a=real(poly(p));
        aNn=a(N+1);
        p=p*Omegac;
        a=real(poly(p));
        aNu=a(N+1);
        b=real(poly(z));
        M=length(b);
        bNn=b(M);
        z=z*Omegac;
        b=real(poly(z));
        bNu=b(M);
        k=k*(aNu/aNn)/(aNn*bNu);
        b0=k;
        b=k*b;
```

10. 椭圆模拟低通滤波器原型设计

```
afd_elip.m
function [b, a]=afd_elip(Wp, Ws, Rp, As);
% Analog Lowpass Filter Design: Elliptic
% --------------------------
% [b, a]=afd_elip(Wp, Ws, Rp, As);
% b=Numberator coefficients of Ha(s)
```

```
% a=Denominator coefficients of Ha(s)
% Wp=Passband edge frequency in rad/sec; Wp>0
% Ws=Stopband edge frequency in rad/sec; Ws>Wp>0
% Rp=Passband ripple in +dB; (Rp>0)
% As=Stopband attenuation in +dB; (As>0)
%
if Wp<=0
        error('Passband edge must be larger than 0')
end
if Ws<=Wp
        error('Stopband edge must be larger than Passband edge')
end
if (Rp<=0)|(As<0)
        error('PB ripple and/or SB attenuation must be larger than 0')
end

ep=sqrt(10^(Rp/10)-1);
A=10^(As/20);
OmegaC=Wp;
k=Wp/Ws;
k1=ep/sqrt(A*A-1);
capk=ellipke([k.^2 1-k.^2]);
capk1=ellipke([k1.^2) 1-(k1.^2)]);
N=ceil(capk(1)*capk1(2)/(capk(2)*capk1(1)));
fprintf('\n * * * Elliptic Filter Order=%2.0f \n', N)
[b, a]=u_elipap(N, Rp, As, OmegaC);
```

11. 冲激不变法模拟到数字滤波器变换

```
imp_invr.m
funtion [b, a]=imp_invr(c, d, T)
% Impulse Invariance Transformation from Analog to Digital Filter
% --------------------------
% [b, a]=imp_invr(c, d, T)
% b=Numerator polynomial in z^(-1) of the digital filter
% a=Denominator polynomial in z^(-1) of the digital filter
% c=Numerator polynomial in s of the analog filter
% d=Denominator polynomial in s of the anglog filter
% T=Sampling (transformation) parameter
%
```

```
[R, p, k]=residue(c, d);
p=exp(p * T);
[b, a]=residuez(R, p, k);
b=real(b'); a=real(a');
```

12. 数字滤波器频率变换

```
zmapping.m
function [bz, az]=zmapping(bZ, aZ, Nz, Dz)
% Frequency band Transformation from Z-domain to z-domain
% -------------------------
% [bz, az]=zmapping(bZ, aZ, Nz, Dz)
% performs:
%           b(z)     b(Z)  |
%           ----=-----     |      N(z)
%           a(z)     z(Z)  |  Z=----
%                          |      D(z)
bzord=(length(bZ)-1 * (length(Nz)-1);
azord=(length(aZ)-1 * (length(Dz)-1);

bz=zeros(1, bzord+1);
for k=0:bzord
    pln=[1];
    for l=0:k-1;
        pln=conv(pln, Nz);
    end
    pld=[1];
    for l=0:bzord-k-1
        pld=conv(pld, Dz);
    end
    bz=bz+bZ(k+1) * conv(pln, pld);
end
az=zeros(1, azord+1);
for k=0:azord
    pln=[1];
    for l=0:k-1
        pln=conv(pln, Nz);
    end
    pld=[1];
    for l=0:azord-k-1
```

```
        pld=conv(pld, Dz);
    end
    az=az+aZ(k+1) * conv(pln, pld);
end
az1=az(1); az=az/az1; bz=bz/az1;
```

13. 利用 Chebyshev Ⅰ型滤波器原型设计高通滤波器

```
cheb1hpf.m
function [b, a]=cheb1hpf(wp, ws, Rp, As);
% IIR Highpass filter design using Chebyshev -1 prototype
% [b, a]=cheb1hpf(wp, ws, Rp, As);
% b=Numberator polynomial of the highpass filter
% a=Denomiantro polynomial of the highpass filter
% wp=Passband frequency in radians
% ws=Stopband frequency in radians
% Rp=Passband ripple in dB
% As=Stopband attenuation in dB
%
% Determine the digital lowpass cutoff frequecies:
wplp=0.2 * pi;
alpha=-(cos((wplp+wp)/2))/(cos((wplp-wp)/2));
wslp=angle(-(exp(-j * ws)+alpha)/(1+alpha * exp(-j * ws)));
%
% Compute Analog lowpass Prototype Specification:
T=1; Fs=1/T;
OmegaP=(2/T) * tan(wplp/2);
OmegaS=(2/T) * tan(wslp/2);
% Design Analog Chebyshev Prototype Lowpass Filter:
[cs, ds]=afd_chb1(OmegaP, OmegaS, Rp, As);
% Perform Bilinear transformation to obtain digital lowpass
[blp, alp]=bilinear(cs, ds, Fs);
% Transform digital lowpass into highpass filter
Nz=-[alpha, 1]; Dz=[1, aplha];
[b, a]=zmapping(blp, alp, Nz, Dz);
```

3.7.1 模拟滤波器的原型特性

例 3.49 设计模拟 Butterworth 低通滤波器

$$\Omega_p = 0.2\pi \qquad R_p = 7\ \text{dB}$$

$$\Omega_s = 0.3\pi \qquad A_s = 16\ \text{dB}$$

解 可直接利用 afd_butt 函数实现。MATLAB 程序为 ex3049.m:

```
% Example 3.49
%
% Butterworth Lowpass Analog filter design
%
Wp=0.2*pi; Ws=0.3*pi; Rp=7; As=16;
Ripple=10 ^ (-Rp/20); Attn=10 ^ (-As/20);
[b, a]=afd_butt(Wp, Ws, Rp, As);
[C, B, A]=sdir2cas(b, a)
[db, mag, pha, w]=freqs_m(b, a, 0.5*pi);
[ha, x, t]=impulse(b, a);
%
% Plots
figure(1)
subplot(2, 2, 1); plot(w/pi, mag); title('Magnitude Response')
xlabel('Analog frequency in pi units'); ylabel('|H|');
axis([0, 0.5, 0, 1.1])
set(gca, 'XTickMode', 'manual', 'XTick', [0, 0.2, 0.3, 0.5]);
set(gca, 'YTickmode', 'manual', 'YTick', [0, Attn, Ripple, 1]); grid
subplot(2, 2, 2); plot(w/pi, db); title('Magnitude in dB')
xlabel('Analog frequency in pi units'); ylabel('decibels');
axis([0, 0.5, -30, 5])
set(gca, 'XTickMode', 'manual', 'XTick', [0, 0.2, 0.3, 0.5]);
set(gca, 'YTickmode', 'manual', 'YTick', [-30, -As, -Rp, 0]); grid
set(gca, 'YTickLabelMode', 'manual', 'YTickLabels', ['30'; '16'; ' 7'; ' 0'])
subplot(2, 2, 3); plot(w/pi, pha/pi); title('Phase Response')
xlabel('Analog frequency in pi units'); ylabel('radians');
axis([0, 0.5, -1, 1])
set(gca, 'XTickMode', 'manual', 'XTick', [0, 0.2, 0.3, 0.5]);
set(gca, 'YTickmode', 'manual', 'YTick', [-1, -0.5, 0, 0.5, 1]); grid
subplot(2, 2, 4); plot(t, ha, [0, max(t)], [0, 0]); title('Impulse Response')
xlabel('time in seconds'); ylabel('ha(t)');
axis([0, max(t), min(ha), max(ha)])
```

执行后得

```
*** Butterworth Filter Order=3
C=
  0.1238
B=
```

```
     0         0         1
A=
     1.0000    0.4985    0.2485
          0    1.0000    0.4985
```

从而得系统函数

$$H_a(s) = \frac{0.1238}{(s^2 + 0.4985s + 0.2485)(s + 0.4985)}$$

同时程序执行得到如图 3.54 所示的 Butterworth 模拟滤波器曲线。

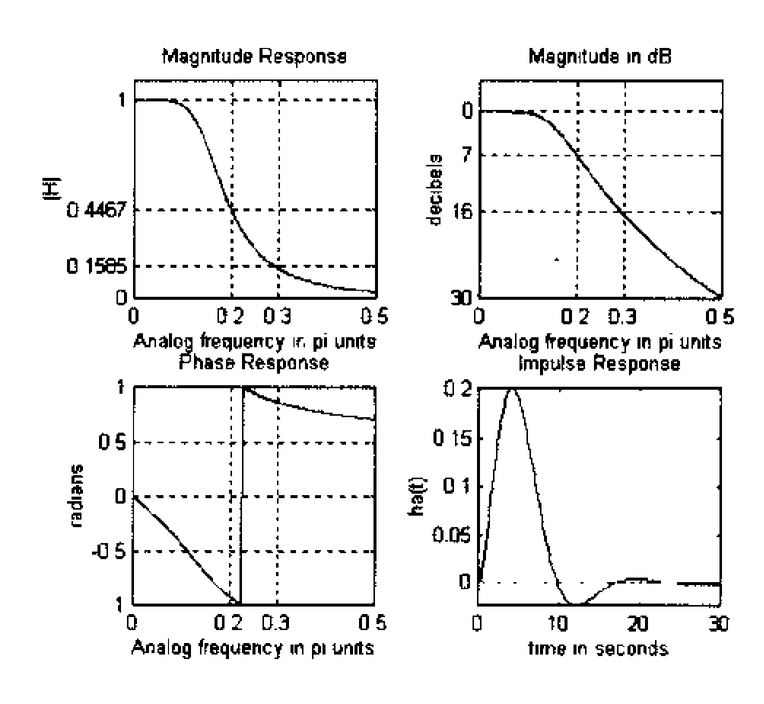


图 3.54 Butterworth 模拟滤波器设计

例 3.50 设计 Chebyshev Ⅰ型模拟低通滤波器

$$\Omega_p = 0.2\pi \qquad R_p = 1\ \text{dB}$$

$$\Omega_s = 0.3\pi \qquad A_s = 16\ \text{dB}$$

解 可直接利用 afd_chb1函数设计，MATLAB 程序为 ex3050.m：

```
% Example 3.50
%
% Chebyshev-1 Lowpass Analog filter design
%
Wp=0.2*pi; Ws=0.3*pi; Rp=1; As=16;
Ripple=10 ^ (-Rp/20); Attn=10 ^ (-As/20);
[b, a]=afd_chb1(Wp, Ws, Rp, As);
[C, B, A]=sdir2cas(b, a)
[db, mag, pha, w]=freqs_m(b, a, 0.5*pi);
[ha, x, t]=impulse(b, a);
%
```

```
% Plots
figure(1);
subplot(2, 2, 1); plot(w/pi, mag); title('Magnitude Response')
xlabel('Analog frequency in pi units'); ylabel(' |H|');
axis([0, 0.5, 0, 1.1])
set(gca, 'XTickMode', 'manual', 'XTick', [0, 0.2, 0.3, 0.5]);
set(gca, 'YTickmode', 'manual', 'YTick', [0, Attn, Ripple, 1]); grid
subplot(2, 2, 2); plot(w/pi, db); title('Magnitude in dB')
xlabel('Analog frequency in pi units'); ylabel('decibels');
axis([0, 0.5, -30, 5])
set(gca, 'XTickMode', 'manual', 'XTick', [0, 0.2, 0.3, 0.5]);
set(gca, 'YTickmode', 'manual', 'YTick', [-30, -As, -Rp, 0]); grid
set(gca, 'YTickLabelMode', 'manual', 'YTickLabels', ['30'; '16'; ' 1'; ' 0'])
subplot(2, 2, 3); plot(w/pi, pha/pi); title('Phase Response')
xlabel('Analog frequency in pi units'); ylabel('radians');
axis([0, 0.5, -1, 1])
set(gca, 'XTickMode', 'manual', 'XTick', [0, 0.2, 0.3, 0.5]);
set(gca, 'YTickmode', 'manual', 'YTick', [-1, -0.5, 0, 0.5, 1]); grid
subplot(2, 2, 4); plot(t, ha, [0, max(t)], [0, 0]); title('Impulse Response')
xlabel('time in seconds'); ylabel('ha(t)');
axis([0, max(t), min(ha), max(ha)])
```

执行后得

```
* * * Chebyshev -1 Filter Order=4
C=
  0.0383
B=
  0    0    1
A=
  1.0000    0.4233    0.1103
  1.0000    0.1753    0.3895
```

从而得系统级联形式为

$$H_a(s) = \frac{0.0383}{(s^2 + 0.4233s + 0.1103)(s^2 + 0.1753s + 0.3895)}$$

执行程序还得到如图 3.55 所示的 Chebyshev Ⅰ型模拟滤波器曲线。

例 3.51 设计椭圆模拟低通滤波器

$$\Omega_p = 0.2\pi \qquad R_p = 1\ \text{dB}$$
$$\Omega_s = 0.3\pi \qquad A_s = 16\ \text{dB}$$

解 可直接利用 afd_elip 函数进行设计，MATLAB 程序为 ex3051.m：

```
% Example 3.51
```

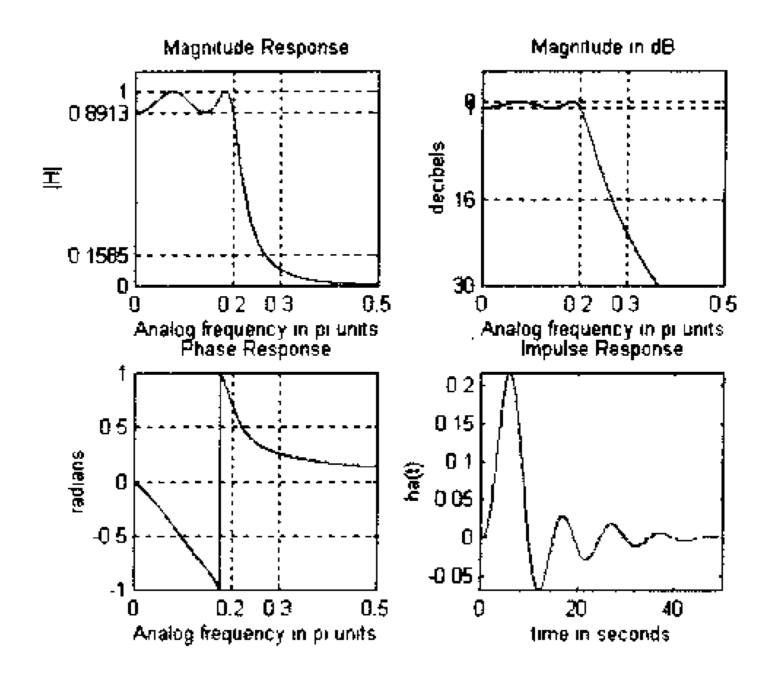


图 3.55 Chebyshev I 型模拟滤波器设计

```
%
% Elliptic Lowpass Analog filter design
%
Wp=0.2 * pi; Ws=0.3 * pi; Rp=1; As=16;
Ripple=10 ^ (-Rp/20); Attn=10 ^ (-As/20);
[b, a]=afd_elip(Wp, Ws, Rp, As);
[C, B, A]=sdir2cas(b, a)
[db, mag, pha, w]=freqs_m(b, a, 0.5 * pi);
[ha, x, t]=impulse(b, a);
%
% Plots
figure(1);
subplot(2, 2, 1); plot(w/pi, mag); title('Magnitude Response')
xlabel('Analog frequency in pi units'); ylabel('|H|');
axis([0, 0.5, 0, 1.1])
set(gca, 'XTickMode', 'manual', 'XTick', [0, 0.2, 0.3, 0.5]);
set(gca, 'YTickmode', 'manual', 'YTick', [0, Attn, Ripple, 1]); grid
subplot(2, 2, 2); plot(w/pi, db); title('Magnitude in dB')
xlabel('Analog frequency in pi units'); ylabel('decibels');
axis([0, 0.5, -30, 1])
set(gca, 'XTickMode', 'manual', 'XTick', [0, 0.2, 0.3, 0.5]);
set(gca, 'YTickmode', 'manual', 'YTick', [-30, -As, -Rp, 0]); grid
set(gca, 'YTickLabelMode', 'manual', 'YTickLabels', ['30'; '16'; ' 1'; ' 0'])
```

```
subplot(2, 2, 3); plot(w/pi, pha/pi); title('Phase Response')
xlabel('Analog frequency in pi units'); ylabel('radians');
axis([0, 0.5, -1, 1])
set(gca, 'XTickMode', 'manual', 'XTick', [0, 0.2, 0.3, 0.5]);
set(gca, 'YTickmode', 'manual', 'YTick', [-1, -0.5, 0, 0.5, 1]); grid
subplot(2, 2, 4); plot(t, ha, [0, max(t)], [0, 0]); title('Impulse Response')
xlabel('time in seconds'); ylabel('ha(t)');
axis([0, max(t), min(ha), max(ha)])
```

执行后得

```
* * * Elliptic Filter Order=3
C=
  0.2740
B=
  1.0000     0     0.6641
A=
  1.0000    0.1696    0.4102
       0    1.0000    0.4435
```

这说明可用 3 阶随圆滤波器实现

$$H_a(s) = \frac{0.274(s^2 + 0.6641)}{(s^2 + 0.1696s + 0.4102)(s + 0.4435)}$$

执行程序同时产生如图 3.56 所示的滤波器曲线。

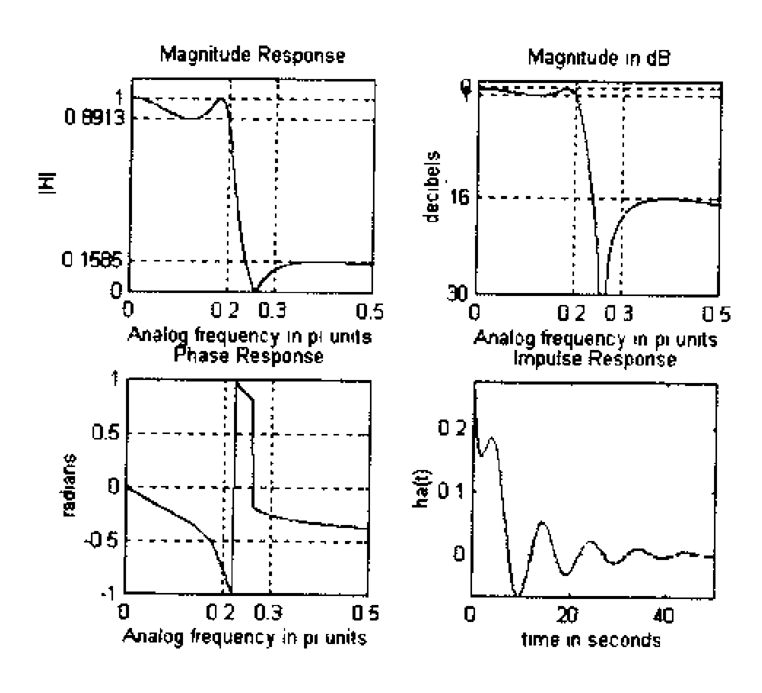


图 3.56 椭圆模拟滤波器设计

例 3.52 利用 Butterworth 原型设计低通滤波器，使满足

$$\omega_p = 0.2\pi \qquad R_p = 1\ \text{dB}$$

$$\omega_s = 0.3\pi \qquad A_s = 15\ \text{dB}$$

解 MATLAB 程序为 ex3052.m：

```
% Example 3.52
%
% Impulse Invariance Transformation
% Butterworth Lowpass Filter Design
%
wp=0.2 * pi;
ws=0.3 * pi;
Rp=1;
As=15;
T=1;
OmegaP=wp * T;
OmegaS=ws * T;
ep=sqrt(10^ (Rp/10)-1);
Ripple=sqrt(1/(1+ep * ep));
Attn=1/(10^ (As/20));
%
% Analog Butterworth Prototype Filter Calculation:
[cs, ds]=afd_butt(OmegaP, OmegaS, Rp, As);
[b, a]=imp_invr(cs, ds, T);
[C, B, A]=dir2par(b, a)
%
% Plot
figure(1);
[db, mag, pha, grd, w]=freqz_m(b, a);
subplot(2, 2, 1); plot(w/pi, mag); title('Magnitude Response')
xlabel('frequency in pi units'); ylabel('|H|'); axis([0, 1, 0, 1.1])
set(gca, 'XTickMode', 'manual', 'XTick', [0, 0.2, 0.3, 1]);
set(gca, 'YTickmode', 'manual', 'YTick', [0, Attn, Ripple, 1]); grid
subplot(2, 2, 3); plot(w/pi, db); title('Magnitude in dB');
xlabel('frequency in pi units'); ylabel('decibels');
axis([0, 1, -40, 5]);
set(gca, 'XTickMode', 'manual', 'XTick', [0, 0.2, 0.3, 1]);
set(gca, 'YTickmode', 'manual', 'YTick', [-50, -15, -1, 0]); grid
set(gca, 'YTickLabelMode', 'manual', 'YTickLabels', ['50'; '15'; ' 1'; ' 0'])
subplot(2, 2, 2); plot(w/pi, pha/pi); title('Phase Response')
xlabel('frequency in pi units'); ylabel('pi units');
axis([0, 1, -1, 1]);
set(gca, 'XTickMode', 'manual', 'XTick', [0, 0.2, 0.3, 1]);
```

```
set(gca, 'YTickmode', 'manual', 'YTick', [-1, 0, 1]); grid
subplot(2, 2, 4); plot(w/pi, grd); title('Group Delay')
xlabel('frequency in pi units'); ylabel('Samples');
axis([0, 1, 0, 10])
set(gca, 'XTickMode', 'manual', 'XTick', [0, 0.2, 0.3, 1]);
set(gca, 'YTickmode', 'manual', 'YTick', [0:2:10]); grid
```

执行后得

```
* * * Butterworth Filter Order=6
C=
  [ ]
B=
    1.8557    -0.6304
   -2.1428     1.1454
    0.2871    -0.4466
A=
    1.0000    -0.9973    0.2570
    1.0000    -1.0691    0.3699
    1.0000    -1.2972    0.6949
```

这说明 6 阶 Butterworth 滤波器的系统函数为

$$H(z) = \frac{1.8587 - 0.6304z^{-1}}{1 - 0.9973z^{-1} + 0.257z^{-2}} + \frac{-2.1428 + 1.1454z^{-1}}{1 - 1.0691z^{-1} + 0.3699z^{-2}} + \frac{0.2871 - 0.4466z^{-1}}{1 - 1.2972z^{-1} + 0.6949z^{-2}}$$

执行程序同时产生如图 3.57 所示的滤波器曲线。

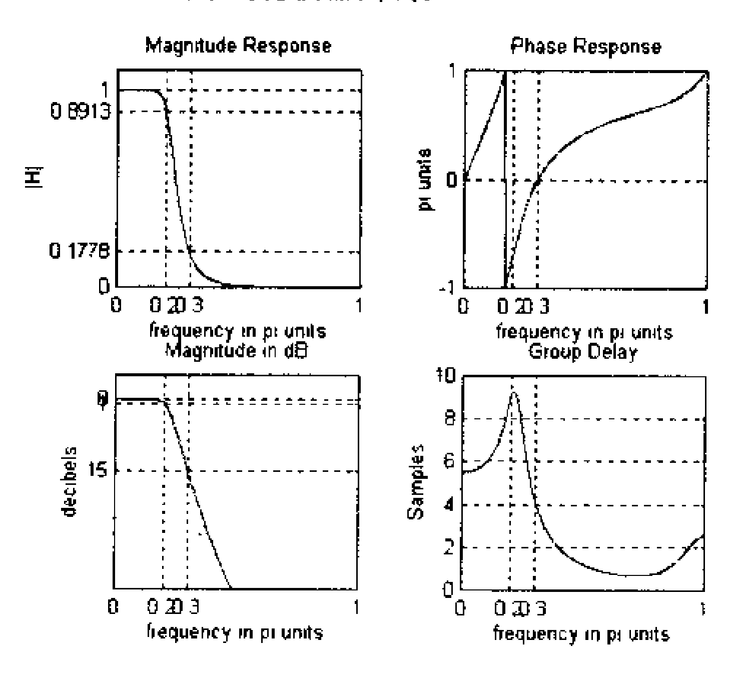


图 3.57 冲激不变法设计数字 Butterworth 低通滤波器

例 3.53 利用 Chebyshev Ⅰ型滤波器原型设计低通滤波器，使之满足

$$\omega_p = 0.2\pi \qquad R_p = 1\ \text{dB}$$
$$\omega_s = 0.3\pi \qquad A_s = 15\ \text{dB}$$

解 MATLAB 程序为 ex3053.m：

```
% Example 3.53
%
% Impulse Invariance Transformation
% Chebyshev-1 Lowpass Filter Design
%
% Digital Filter Specifications:
wp=0.2*pi;
ws=0.3*pi;
Rp=1;
As=15;
T=1;
OmegaP=wp * T;
OmegaS=ws * T;
ep=sqrt(10^(Rp/10)-1);
Ripple=sqrt(1/(1+ep*ep));
Attn=1/(10^(As/20));        % Stopband Attenuation
[cs, ds]=afd_chb1(OmegaP, OmegaS, Rp, As);
[b, a]=imp_invr(cs, ds, T);
[C, B, A]=dir2par(b, a)
%
% Plot
figure(1);
[db, mag, pha, grd, w]=freqz_m(b, a);
subplot(2, 2, 1); plot(w/pi, mag); title('Magnitude Response')
xlabel('frequency in pi units'); ylabel('|H|'); axis([0, 1, 0, 1.1])
set(gca, 'XTickMode', 'manual', 'XTick', [0, 0.2, 0.3, 1]);
set(gca, 'YTickmode', 'manual', 'YTick', [0, Attn, Ripple, 1]); grid
subplot(2, 2, 3); plot(w/pi, db); title('Magnitude in dB');
xlabel('frequency in pi units'); ylabel('decibels');
axis([0, 1, -40, 5]);
set(gca, 'XTickMode', 'manual', 'XTick', [0, 0.2, 0.3, 1]);
set(gca, 'YTickmode', 'manual', 'YTick', [-50, -15, -1, 0]); grid
set(gca, 'YTickLabelMode', 'manual', 'YTickLabels', ['50'; '15'; ' 1'; ' 0'])
subplot(2, 2, 2); plot(w/pi, pha/pi); title('Phase Response')
xlabel('frequency in pi units'); ylabel('pi units');
```

```
axis([0, 1, -1, 1]);
set(gca, 'XTickMode', 'manual', 'XTick', [0, 0.2, 0.3, 1]);
set(gca, 'YTickmode', 'manual', 'YTick', [-1, 0, 1]); grid
subplot(2, 2, 4); plot(w/pi, grd); title('Group Delay')
xlabel('frequency in pi units'); ylabel('Samples');
axis([0, 1, 0, 15])
set(gca, 'XTickMode', 'manual', 'XTick', [0, 0.2, 0.3, 1]);
set(gca, 'YTickmode', 'manual', 'YTick', [0:5:15]); grid
```

执行后得

```
* * * Chebyshev-1 Filter Order=4
C=
  [ ]
B=
  -0.0833    -0.0246
   0.0833     0.0239
A=
   1.0000    -1.4934    0.8392
   1.0000    -1.5658    0.6549
```

这表明 4 阶 Chebyshev Ⅰ型滤波器的系统函数为

$$H(z)=\frac{-0.0833-0.0246z^{-1}}{1-1.4934z^{-1}+0.8392z^{-2}}+\frac{0.083+0.0239z^{-1}}{1-1.5658z^{-1}+0.6549z^{-2}}$$

执行程序同时还得到了如图 3.58 所示的滤波器曲线。

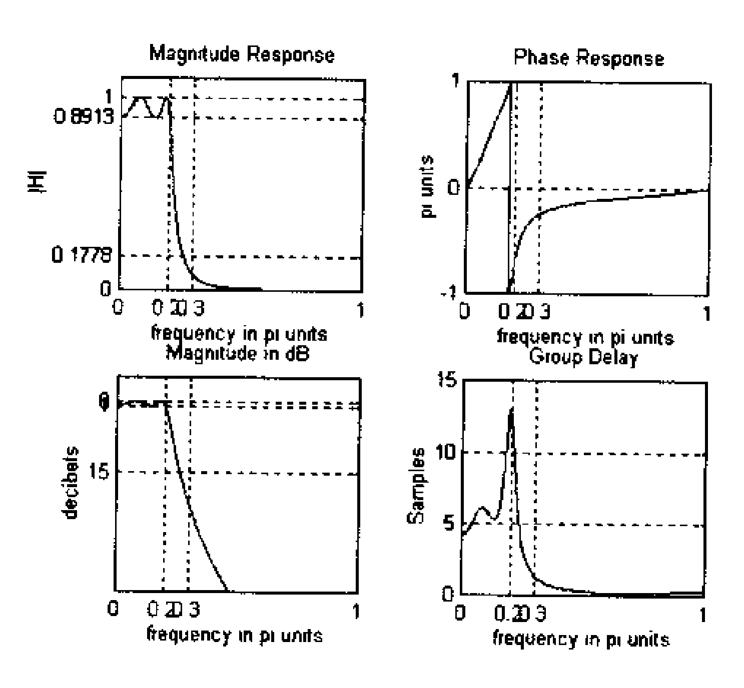


图 3.58 冲激不变法设计数字 Chebyshev Ⅰ型低通滤波器

例 3.54 利用椭圆滤波器原型设计低通数字滤波器，使之满足

$$\omega_p = 0.2\pi \qquad R_p = 1\ \text{dB}$$
$$\omega_s = 0.3\pi \qquad A_s = 15\ \text{dB}$$

解 MATLAB 程序为 ex3054.m：

```
% Example 3.54
%
% Impulse Invariance Transformation
% Elliptic Lowpass Filter Design
%
% Digital Filter Specifications:
wp=0.2*pi;
ws=0.3*pi;
Rp=1;
As=15;
T=1;
OmegaP=wp * T;
OmegaS=ws * T;
ep=sqrt(10^(Rp/10)-1);
Ripple=sqrt(1/(1+ep*ep));
Attn=1/(10^(As/20));                    % Stopband Attenuation
[cs, ds]=afd_elip(OmegaP, OmegaS, Rp, As);
[b, a]=imp_invr(cs, ds, T);
[C, B, A]=dir2par(b, a)
%
% Plot
figure(1);
[db, mag, pha, grd, w]=freqz_m(b, a);
subplot(2, 2, 1); plot(w/pi, mag); title('Magnitude Response')
xlabel('frequency in pi units'); ylabel('|H|'); axis([0, 1, 0, 1.1])
set(gca, 'XTickMode', 'manual', 'XTick', [0, 0.2, 0.3, 1]);
set(gca, 'YTickmode', 'manual', 'YTick', [0, Attn, Ripple, 1]); grid
subplot(2, 2, 3); plot(w/pi, db); title('Magnitude in dB');
xlabel('frequency in pi units'); ylabel('decibels');
axis([0, 1, -40, 5]);
set(gca, 'XTickMode', 'manual', 'XTick', [0, 0.2, 0.3, 1]);
set(gca, 'YTickmode', 'manual', 'YTick', [-50, -15, -1, 0]); grid
set(gca, 'YTickLabelMode', 'manual', 'YTickLabels', ['50'; '15'; ' 1'; ' 0'])
subplot(2, 2, 2); plot(w/pi, pha/pi); title('Phase Response')
xlabel('frequency in pi units'); ylabel('pi units');
axis([0, 1, -1, 1]);
```

```
set(gca, 'XTickMode', 'manual', 'XTick', [0, 0.2, 0.3, 1]);
set(gca, 'YTickmode', 'manual', 'YTick', [-1, 0, 1]); grid
subplot(2, 2, 4); plot(w/pi, grd); title('Group Delay')
xlabel('frequency in pi units'); ylabel('Samples');
axis([0, 1, 0, 15])
set(gca, 'XTickMode', 'manual', 'XTick', [0, 0.2, 0.3, 1]);
set(gca, 'YTickmode', 'manual', 'YTick', [0:5:15]); grid
```

执行后得

```
* * * Elliptic Filter Order=3
C=
  [ ]
B=
  -0.1600     0.1299
   0.4560     0
A=
   1.0000    -1.4854    0.8521
   1.0000    -0.6338    0
```

这表明 3 阶椭圆滤波器的系统函数为

$$H(z)=\frac{-0.16+0.1299z^{-1}}{1-1.4854z^{-1}+0.8521z^{-2}}+\frac{0.4560}{1-0.6338z^{-1}}$$

执行程序还得到了如图 3.59 所示的滤波器曲线。

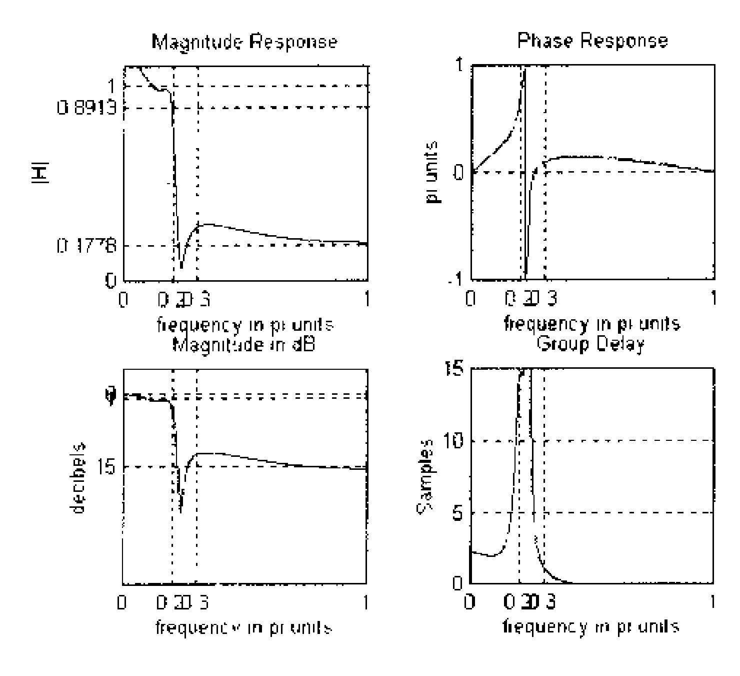


图 3.59 冲激不变法设计椭圆低通滤波器

例 3.55 与例 3.51 类似，利用双线性变换设计数字 Butterworth 滤波器

$$\omega_p=0.2\pi \qquad R_p=1\ \text{dB}$$

$$\omega_s = 0.3\pi \qquad A_s = 15\ \mathrm{dB}$$

解 MATLAB 程序为 ex3055.m：

```
% Example 3.55
%
% BiLinear Transformation:
% Butterworth Lowpass Filter Design
%
% Digital Filter Specifications:
wp=0.2*pi;
ws=0.3*pi;
Rp=1;
As=15;
T=1; Fs=1/T;
OmegaP=(2/T)*tan(wp/2);
OmegaS=(2/T)*tan(ws/2);
ep=sqrt(10^(Rp/10)-1);
Ripple=sqrt(1/(1+ep*ep));
Attn=1/(10^(As/20));          % Stopband Attenuation
[cs, ds]=afd_butt(OmegaP, OmegaS, Rp, As);
[b, a]=bilinear(cs, ds, T);
[C, B, A]=dir2cas(b, a);
%
% Plot
figure(1);
[db, mag, pha, grd, w]=freqz_m(b, a);
subplot(2, 2, 1); plot(w/pi, mag); title('Magnitude Response')
xlabel('frequency in pi units'); ylabel('|H|'); axis([0, 1, 0, 1.1])
set(gca, 'XTickMode', 'manual', 'XTick', [0, 0.2, 0.3, 1]);
set(gca, 'YTickmode', 'manual', 'YTick', [0, Attn, Ripple, 1]); grid
subplot(2, 2, 3); plot(w/pi, db); title('Magnitude in dB');
xlabel('frequency in pi units'); ylabel('decibels');
axis([0, 1, -40, 5]);
set(gca, 'XTickMode', 'manual', 'XTick', [0, 0.2, 0.3, 1]);
set(gca, 'YTickmode', 'manual', 'YTick', [-50, -15, -1, 0]); grid
set(gca, 'YTickLabelMode', 'manual', 'YTickLabels', ['50'; '15'; ' 1'; ' 0'])
subplot(2, 2, 2); plot(w/pi, pha/pi); title('Phase Response')
xlabel('frequency in pi units'); ylabel('pi units');
axis([0, 1, -1, 1]);
set(gca, 'XTickMode', 'manual', 'XTick', [0, 0.2, 0.3, 1]);
```

```
set(gca, 'YTickmode', 'manual', 'YTick', [-1, 0, 1]); grid
subplot(2, 2, 4); plot(w/pi, grd); title('Group Delay')
xlabel('frequency in pi units'); ylabel('Samples');
axis([0, 1, 0, 10])
set(gca, 'XTickMode', 'manual', 'XTick', [0, 0.2, 0.3, 1]);
set(gca, 'YTickmode', 'manual', 'YTick', [0:2:10]); grid
```

执行后得

```
* * * Butterworth Filter Order=6
```

同时产生如图 3.60 所示的滤波器曲线。

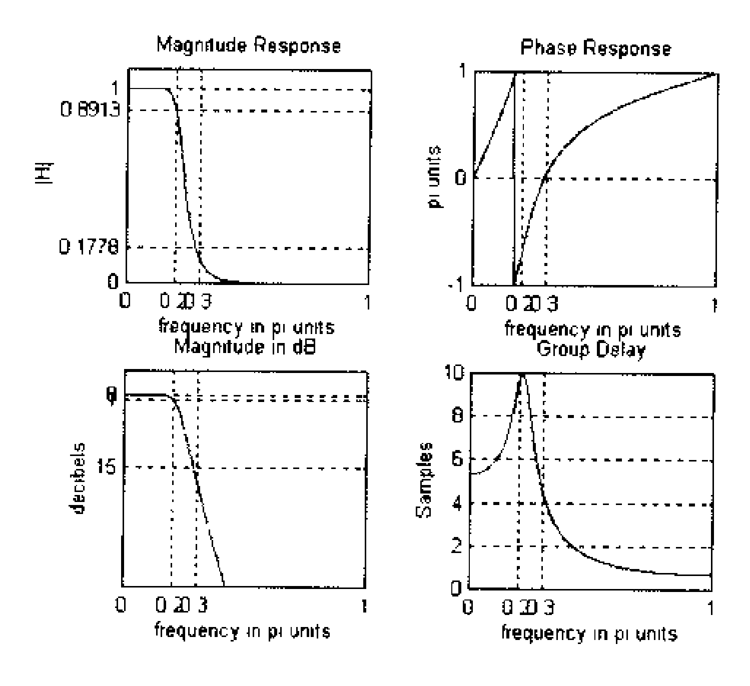


图 3.60 双线性变换法设计 Butterworth 低通滤波器

例 3.56 与例 3.52 类似，利用双线性变换设计数字 Chebyshev **I** 型滤波器

$$\omega_p = 0.2\pi \qquad R_p = 1\ \text{dB}$$
$$\omega_s = 0.3\pi \qquad A_s = 15\ \text{dB}$$

解 MATLAB 程序为 ex3056.m：

```
% Example 3.56
%
% BiLinear Transformation:
% Chebyshev-2 Lowpass Filter Design
%
% Digital Filter Specifications:
wp=0.2*pi;
ws=0.3*pi;
Rp=1;
As=15;
T=1; Fs=1/T;
```

```
OmegaP=(2/T) * tan(wp/2);
OmegaS=(2/T) * tan(ws/2);
ep=sqrt(10^ (Rp/10)-1);
Ripple=sqrt(1/(1+ep * ep));
Attn=1/(10^ (As/20));                % Stopband Attenuation
[cs, ds]=afd_chb2(OmegaP, OmegaS, Rp, As);
[b, a]=bilinear(cs, ds, T);
[C, B, A]=dir2cas(b, a);
%
% Plot
figure(1);
[db, mag, pha, grd, w]=freqz_m(b, a);
subplot(2, 2, 1); plot(w/pi, mag); title('Magnitude Response')
xlabel('frequency in pi units'); ylabel(' |H|');
axis([0, 1, 0, 1.1])
set(gca, 'XTickMode', 'manual', 'XTick', [0, 0.2, 0.3, 1]);
set(gca, 'YTickmode', 'manual', 'YTick', [0, Attn, Ripple, 1]); grid
subplot(2, 2, 3); plot(w/pi, db); title('Magnitude in dB');
xlabel('frequency in pi units'); ylabel('decibels');
axis([0, 1, -40, 5]);
set(gca, 'XTickMode', 'manual', 'XTick', [0, 0.2, 0.3, 1]);
set(gca, 'YTickmode', 'manual', 'YTick', [-50, -15, -1, 0]); grid
set(gca, 'YTickLabelMode', 'manual', 'YTickLabels', ['50'; '15'; ' 1'; ' 0'])
subplot(2, 2, 2); plot(w/pi, pha/pi); title('Phase Response')
xlabel('frequency in pi units'); ylabel('pi units');
axis([0, 1, -1, 1]);
set(gca, 'XTickMode', 'manual', 'XTick', [0, 0.2, 0.3, 1]);
set(gca, 'YTickmode', 'manual', 'YTick', [-1, 0, 1]); grid
subplot(2, 2, 4); plot(w/pi, grd); title('Group Delay')
xlabel('frequency in pi units'); ylabel('Samples');
axis([0, 1, 0, 15])
set(gca, 'XTickMode', 'manual', 'XTick', [0, 0.2, 0.3, 1]);
set(gca, 'YTickmode', 'manual', 'YTick', [0:5:15]); grid
```

执行后得

```
* * * Chebyshev-2 Filter Order=4
```

同时还得到了如图 3.61 所示的滤波器曲线。

例 3.57 与例 3.53 类似，利用双线性变换设计数字椭圆滤波器

$$\omega_p = 0.2\pi \qquad R_p = 1\ \text{dB}$$

$$\omega_s = 0.3\pi \qquad A_s = 15\ \text{dB}$$

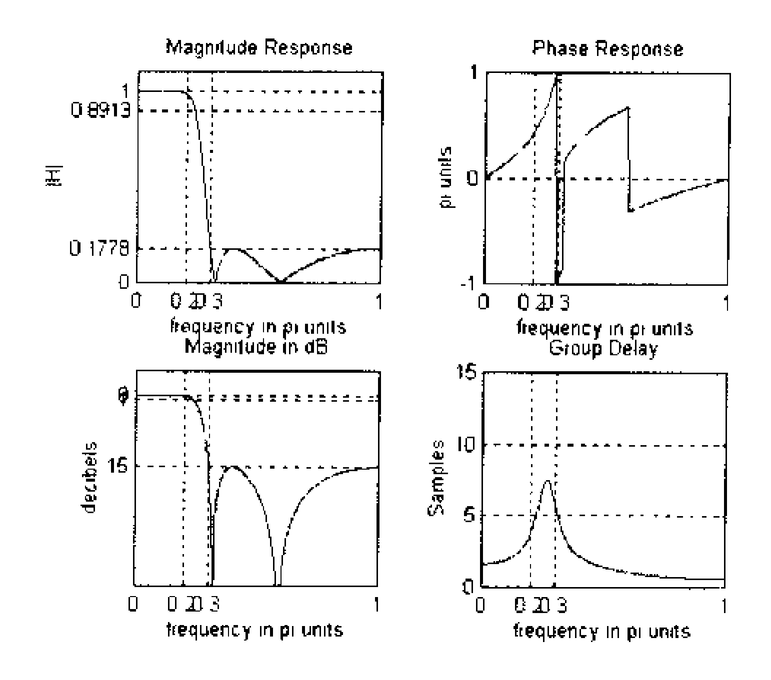


图 3.61　双线性变换法设计 Chebyshev Ⅱ 型低通滤波器

解　MATLAB 程序为 ex3057.m：

```
% Example 3.57
%
% BiLinear Transformation:
% Elliptic Lowpass Filter Design
%
% Digital Filter Specifications:
wp=0.2*pi;
ws=0.3*pi;
Rp=1;
As=15;
T=1; Fs=1/T;
OmegaP=(2/T)*tan(wp/2);
OmegaS=(2/T)*tan(ws/2);
ep=sqrt(10^(Rp/10)-1);
Ripple=sqrt(1/(1+ep*ep));
Attn=1/(10^(As/20));                    % Stopband Attenuation
[cs, ds]=afd_elip(OmegaP, OmegaS, Rp, As);
[b, a]=bilinear(cs, ds, T);
[C, B, A]=dir2cas(b, a);
%
% Plot
figure(1);
[db, mag, pha, grd, w]=freqz_m(b, a);
```

```
subplot(2, 2, 1); plot(w/pi, mag); title('Magnitude Response')
xlabel('frequency in pi units'); ylabel('|H|');
axis([0, 1, 0, 1.1])
set(gca, 'XTickMode', 'manual', 'XTick', [0, 0.2, 0.3, 1]);
set(gca, 'YTickmode', 'manual', 'YTick', [0, Attn, Ripple, 1]); grid
subplot(2, 2, 3); plot(w/pi, db); title('Magnitude in dB');
xlabel('frequency in pi units'); ylabel('decibels');
axis([0, 1, -40, 5]);
set(gca, 'XTickMode', 'manual', 'XTick', [0, 0.2, 0.3, 1]);
set(gca, 'YTickmode', 'manual', 'YTick', [-50, -15, -1, 0]); grid
set(gca, 'YTickLabelMode', 'manual', 'YTickLabels', ['50'; '15'; ' 1'; ' 0'])
subplot(2, 2, 2); plot(w/pi, pha/pi); title('Phase Response')
xlabel('frequency in pi units'); ylabel('pi units');
axis([0, 1, -1, 1]);
set(gca, 'XTickMode', 'manual', 'XTick', [0, 0.2, 0.3, 1]);
set(gca, 'YTickmode', 'manual', 'YTick', [-1, 0, 1]); grid
subplot(2, 2, 4); plot(w/pi, grd); title('Group Delay')
xlabel('frequency in pi units'); ylabel('Samples');
axis([0, 1, 0, 15])
set(gca, 'XTickMode', 'manual', 'XTick', [0, 0.2, 0.3, 1]);
set(gca, 'YTickmode', 'manual', 'YTick', [0:5:15]); grid
```

执行后得

```
* * * Elliptic Filter Order=3
```

同时还得到如图 3.62 所示的滤波器曲线。

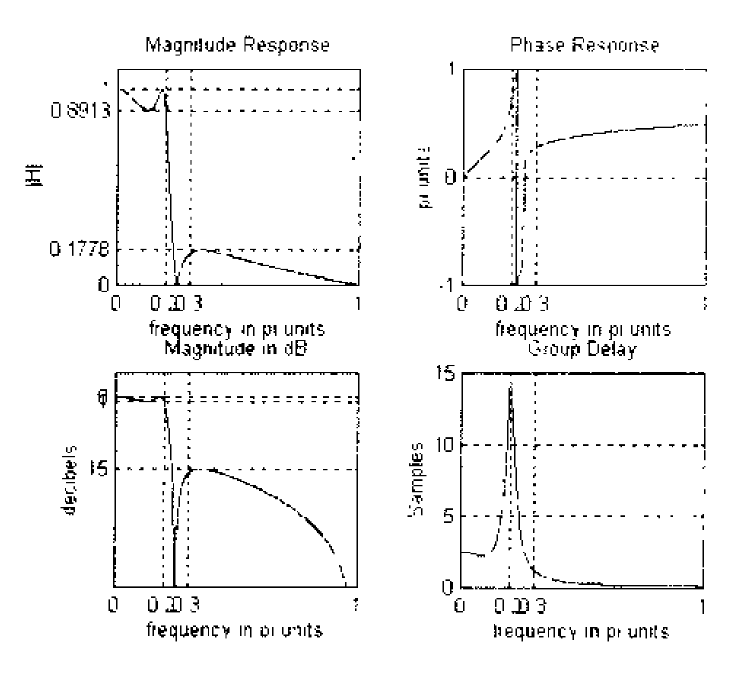


图 3.62 双线性变换设计椭圆低通滤波器

3.7.3 低通滤波器设计

例 3.58 设计数字 Butterworth 低通滤波器

$$\omega_p = 0.2\pi \qquad R_p = 1\ \text{dB}$$
$$\omega_s = 0.3\pi \qquad A_s = 15\ \text{dB}$$

解 MATLAB 程序为 ex3058.m：

```
% Example 3.58
%
% Butterworth Lowpass Filter Design:
% Use of the BUTTER function
%
% Digital Filter Specifications:
wp=0.2*pi;
ws=0.3*pi;
Rp=1;
As=15;
T=1;
OmegaP=(2/T)*tan(wp/2);
OmegaS=(2/T)*tan(ws/2);
[N, wn]=buttord(OmegaP, OmegaS, Rp, As, 's')
wn=wn/pi;
[b, a]=butter(N, wn);
[C, B, A]=dir2cas(b, a)
```

执行后得

```
N=
  6
wn=
  0.7662
C=
  9.2957e-004
B=
  1.0000     2.0105    1.0106
  1.0000     1.9894    0.9895
  1.0000     2.0001    1.0000
A=
  1.0000    -0.8630    0.1978
  1.0000    -0.9669    0.3420
  1.0000    -1.2218    0.6957
```

例 3.59 设计数字 Chebyshev Ⅰ型低通滤波器

$$\omega_p = 0.2\pi \qquad R_p = 1\ \text{dB}$$
$$\omega_s = 0.3\pi \qquad A_s = 15\ \text{dB}$$

解 MATLAB 程序为 ex3059. m：

```
% Example 3. 59
%
% Butterworth Lowpass Filter Design:
% Use of the CHEBY1 function
%
% Digital Filter Specifications:
wp=0.2 * pi;
ws=0.3 * pi;
Rp=1;
As=15;
T=1;
OmegaP=(2/T) * tan(wp/2);
OmegaS=(2/T) * tan(ws/2);
[N, wn]=cheb1ord(OmegaP, OmegaS, Rp, As, 's')
wn=wn/pi;
[b, a]=cheby1(N, Rp, wn);
[C, B, A]=dir2cas(b, a)
```

执行后得

```
N=
  4
wn=
  0.6498
C=
  0.0021
B=
  1.0000    2.0024    1.0024
  1.0000    1.9976    0.9976
A=
  1.0000   -1.4728    0.8441
  1.0000   -1.5386    0.6394
```

例 3.60 先设计 Chebyshev Ⅰ型低通滤波器

$$\omega_p' = 0.2\pi \qquad R_p = 1\ \text{dB}$$
$$\omega_s' = 0.3\pi \qquad A_s = 15\ \text{dB}$$

然后设计高通滤波器，使波纹系数同上，截止截率为 $\omega_p=0.6\pi$。

解 MATLAB 程序为 ex3060. m：

```
% Example 3. 60
```

```
%
% Chebyshev-1 Highpass Filter Design:
% Use of the ZMAPPING function
%
% Digital Lowpass Filter Specifications:
%
wplp=0.2 * pi;
wslp=0.3 * pi;
Rp=1;
As=15;
T=1; Fs=1/T;
OmegaP=(2/T) * tan(wplp/2);
OmegaS=(2/T) * tan(wslp/2);
ep=sqrt(10^(Rp/10)-1);
Ripple=sqrt(1/(1+ep * ep));
Attn=1/(10^(As/20));
[cs, ds]=afd_chb1(OmegaP, OmegaS, Rp, As);
[blp, alp]=bilinear(cs, ds, T);
wphp=0.6 * pi;
% LP-to-HP frequency-band transformation:
alpha=-(cos((wplp+wphp)/2))/(cos((wplp-wphp)/2))
Nz=-[alpha, 1]; Dz=[1, alpha];
[bhp, ahp]=zmapping(blp, alp, Nz, Dz);
[C, B, A]=dir2cas(bhp, ahp)
%
% Plot
figure(1); subplot(1, 1, 1)
[dbl, magl, phal, grdl, w]=freqz_m(blp, alp);
subplot(2, 2, 1); plot(w/pi, magl);
title('Lowpass Filter Magnitude Response')
xlabel('frequency in pi units'); ylabel('|H|'); axis([0, 1, 0, 1]);
set(gca, 'XTickMode', 'manual', 'XTick', [0, 0.2, 1]);
set(gca, 'YTickMode', 'manual', 'YTick', [0, Ripple, 1]); grid
subplot(2, 2, 2); plot(w/pi, dbl);
title('Lowpass Filter Magnitude in dB');
xlabel('frequency in pi units'); ylabel('decibels');
axis([0 1 -30 0]);
set(gca, 'XTickMode', 'manual', 'XTick', [0, 0.2, 1])
set(gca, 'YTickMode', 'manual', 'YTick', [-30, -Rp, 0]);
```

```
set(gca, 'YTickLabelMode', 'manual', 'YTickLabels', ['30'; ' 1'; ' 0']);
[dbh, magh, phah, grdh, w]=freqz_m(bhp, ahp);
subplot(2, 2, 3); plot(w/pi, magh);
title('Highpass Filter Magnitude Response')
xlabel('frequency in pi units'); ylabel(' |H|'); axis([0, 1, 0, 1])
set(gca, 'XTickMode', 'manual', 'XTick', [0, 0.6, 1]);
set(gca, 'YTickMode', 'manual', 'YTick', [0, Ripple, 1]); grid
subplot(2, 2, 4); plot(w/pi, dbh);
title('Highpass Filter Magnitude in dB');
xlabel('frequency in pi units'); ylabel('decibels');
axis([0 1 -30 0]);
set(gca, 'XTickMode', 'manual', 'XTick', [0, 0.6, 1])
set(gca, 'YTickMode', 'manual', 'YTick', [-30, -Rp, 0]); grid
set(gca, 'YTickLabelMode', 'manual', 'YTickLabels', ['30'; ' 1'; ' 0']);
```

执行后得

```
* * * Chebyshev-1 Filter Order=4
alpha=
  -0.3820
C=
  0.0243
B=
  1.0000    -2.0000    1.0000
  1.0000    -2.0000    1.0000
A=
  1.0000    1.0416     0.4019
  1.0000    0.5561     0.7647
```

因此高通滤波器的系统函数为

$$H(z) = \frac{0.0243(1-z^{-1})^4}{(1+1.0416z^{-1}+0.4019z^{-2})(1+0.5561z^{-1}+0.7647z^{-2})}$$

同时得到如图 3.63 所示的滤波器曲线。

例 3.61 利用 Chebyshev Ⅰ型滤波器原型，设计高通数字滤波器

$$\omega_p = 0.6\pi \qquad R_p = 1\ \text{dB}$$
$$\omega_s = 0.4586 \qquad A_s = 15\ \text{dB}$$

解 MATLAB 程序为 ex3061.m：

```
% Example 3.61
%
% Chebyshev-1 Highpass Filter Design:
% Use of the CHEBY1HPF function
%
```

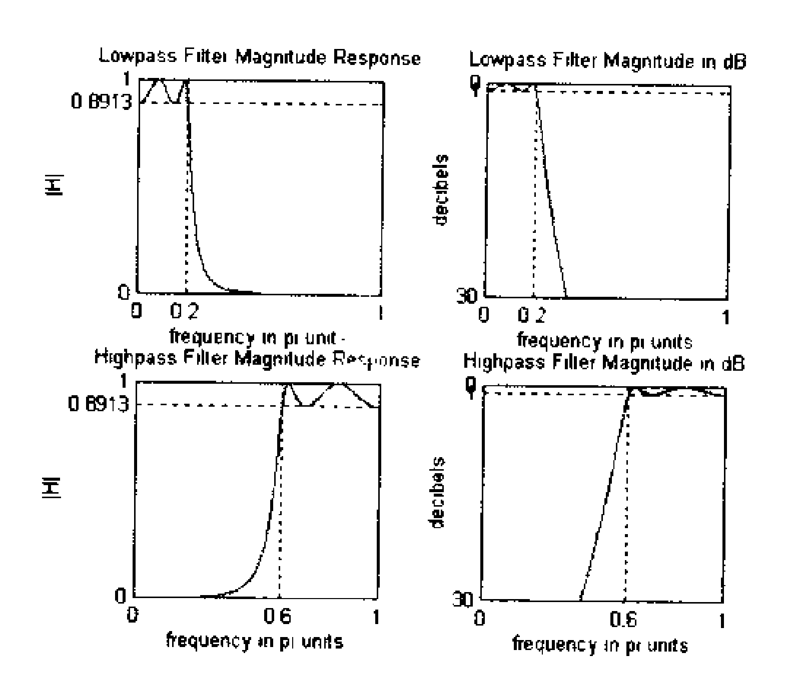


图 3.63 低通和高通滤波器特性

```
% Digital Highpass Filter Specifications:
wp=0.6*pi;
ws=0.4586*pi;
Rp=1;
As=15;
[b,a]=cheb1hpf(wp,ws,Rp,As);
[C,B,A]=dir2cas(b,a)
```

执行结果与例 3.59 一样，这说明这两种方法是一致的，显然这里的程序要短得多。

附录 A

MATLAB 命令参考

MATLAB 系统提供近 20 类基本命令函数，它们有一部分是 MATLAB 的内部命令，有一部分是以 M 文件形式出现的函数，这些 M 文件按类归于一子目录下，每个目录中除了以 M 文件表示的函数命令之外，还有一个特殊的文件 contents. m，它包含了该目录各个 M 文件的简介。每个函数文件中都包含了这一函数的用法指南，因此可用命令：

help fn

来显示有关函数 fn 的帮助信息(fn 为 M 文件名)，也可用命令：

help dn

来显示该目录下各函数文件的简要说明(dn 为目录名)。

限于篇幅，本附录不再列出各个函数详细说明，用户可利用 help 命令获得这些信息，也可参看《MATLAB 程序设计语言》一书。

表 A. 1 为 20 类基本命令函数的子目录及其含义，表 A. 2～A. 20 中列出各类函数的简要说明，以供用户参考。

表 A. 1　基本命令函数目录

目　录　名	命　令　函　数	索　　引
general	通用命令	表 A. 2
ops	操作符和特殊字符	表 A. 3
elfun	基本数学函数	表 A. 4
specfun	特殊数学函数	表 A. 5
elmat	基本矩阵和矩阵操作	表 A. 6
specmat	特殊矩阵	表 A. 7
matfun	矩阵函数——数值线性代数	表 A. 8
sparfun	稀疏矩阵函数	表 A. 9
datafun	数据分析和傅里叶变换函数	表 A. 10
funfun	泛函——非线性数值方法	表 A. 11
polyfun	多项式和内插函数	表 A. 12
graphics	通用图形函数	表 A. 13
plotxy	二维图形函数	表 A. 14
plotxyz	三维图形函数	表 A. 15

续表

目 录 名	命 令 函 数	索 引
lang	语言结构和调试	表 A.16
color	颜色控制和亮度模型函数	表 A.17
strfun	字符串函数	表 A.18
sounds	音频处理函数	表 A.19
iofun	低级文件 I/O 函数	表 A.20
demos	演示例子	—

表 A.2 通用命令

■ 管理命令和函数		
	help	在线帮助文本
	doc	装入超文本说明
	what	M、MAT、MEX 文件的目录列表
	type	列出 M 文件
	lookfor	通过 help 条目搜索关键字
	which	定位函数和文件
	demo	运行演示程序
	path	控制 MATLAB 的搜索路径
■ 管理变量和工作空间		
	who	列出当前变量
	whos	列出当前变量(长表)
	load	从磁盘文件中恢复变量
	save	保存工作空间变量
	clear	从内存中清除变量和函数
	pack	整理工作空间内存
	size	矩阵的尺寸
	length	向量的长度
	disp	显示矩阵或文本
■ 与文件和操作系统有关的命令		
	cd	改变当前工作目录
	dir	目录列表
	delete	删除文件

续表

■ 与文件和操作系统有关的命令		
	getenv	获取环境变量值
	!	执行操作系统命令
	unix	执行操作系统命令并返回结果
	diary	保存 MATLAB 任务
■ 控制命令窗口		
	cedit	设置命令行编辑
	clc	清命令窗口
	home	光标置左上角
	format	设置输出格式
	echo	MATLAB 文件内使用的回显命令
	more	在命令窗口中控制分页输出
■ 启动和退出 MATLAB		
	quit	退出 MATLAB
	startup	引用 MATLAB 时所执行的 M 文件
	matlabrc	主启动 M 文件
■ 一般信息		
	info	MATLAB 系统信息及 Mathworks 公司信息
	subscribe	成为 MATLAB 的订购用户
	hostid	MATLAB 主服务程序的识别代号
	whatsnew	在说明书中未包含的新信息
	ver	版本信息

表 A.3　操作符和特殊字符

■ 操作符和特殊字符		
	+	加
	−	减
	*	矩阵乘法
	.*	数组乘法
	^	矩阵幂
	.^	数组幂
	\	左除或反斜杠

续表

■ **操作符和特殊字符**		
	/	右除或斜杠
	./	数组除
	kron	Kronecker 张量积
	:	冒号
	()	圆括号
	[]	方括号
	.	小数点
	..	父目录
	...	继续
	,	逗号
	;	分号
	%	注释
	!	感叹号
	'	转置或引用
	=	赋值
	==	相等
	< >	关系操作符
	&	逻辑与
	|	逻辑或
	~	逻辑非
	xor	逻辑异或
■ **逻辑函数**		
	exist	检查变量或函数是否存在
	any	向量的任一元为真，则其值为真
	all	向量的所有元为真，则其值为真
	find	找出非零元素的索引号
	isnan	当含 NaN 时，其值为真
	isinf	当含无限大元时，其值为真
	finite	当含有限值元时，其值为真
	isempt	当矩阵为空矩阵时，其值为真
	isreal	当矩阵为实矩阵时，其值为真
	issparse	当矩阵为稀疏矩阵时，其值为真
	isstr	当矩阵为文本串时，其值为真
	isglobal	当变量为全局变量时，其值为真

表 A.4　基本数学函数

■ 三角函数		
	sin	正弦
	sinh	双曲正弦
	asin	反正弦
	asinh	反双曲正弦
	cos	余弦
	cosh	双曲余弦
	acos	反余弦
	acosh	反双曲余弦
	tan	正切
	tanh	双曲正切
	atan	反正切
	atan2	四象限反正切
	atanh	反双曲正切
	sec	正割
	sech	双曲正割
	asec	反正割
	asech	反双曲正割
	csc	余割
	csch	双曲余割
	acsc	反余割
	acsch	反双曲余割
	cot	余切
	coth	双曲余切
	acot	反余切
	acoth	反双曲余切
■ 指数函数		
	exp	指数
	log	自然对数
	log10	常用对数
	sqrt	平方根

续表

■ 复数函数		
	abs	绝对值
	argle	相角
	conj	复共轭
	image	复数虚部
	real	复数实部
■ 数值函数		
	fix	朝零方向取整
	floor	朝负无穷大方向取整
	ceil	朝正无穷大方向取整
	round	朝最近的整数取整
	rem	除后余数
	sign	符号函数

表 A.5 特殊数学函数

	besselj	第一类 Bessel(贝塞尔)函数
	bessely	第二类 Bessel 函数
	besseli	改进的第一类 Bessel 函数
	besselk	改进的第二类 Bessel 函数
	beta	β 类函数
	betainc	非完全的 β 函数
	betaln	β 函数的对数
	ellipj	雅可比椭圆函数
	ellipke	完全椭圆积分
	erf	误差函数
	erfc	互补误差函数
	erfcx	比例互补误差函数
	erfinv	逆误差函数
	expint	指数积分函数
	gamma	γ 函数
	gcd	最大公约数
	gammainc	非完全 γ 函数

续表

	lcm	最小公倍数
	log2	分割浮点数
	pow2	比例浮点数
	rat	有理逼近
	rats	有理输出
	cart2pol	变卡笛尔坐标为极坐标
	cart2sph	变卡笛尔坐标为球坐标
	pol2cart	变极坐标为卡笛尔坐标
	sph2cart	变球坐标为卡笛尔坐标

表 A.6　基本矩阵和矩阵操作

■ 基本矩阵		
	zeros	零矩阵
	ones	全"1"矩阵
	eye	单位矩阵
	rand	均匀分布的随机数矩阵
	randn	正态分布的随机数矩阵
	linspace	线性间隔的向量
	logspace	对数间隔的向量
	meshgrid	三维图形的 X 和 Y 数组
	:	规则间隔的向量
■ 特殊变量和常数		
	ans	当前的答案
	eps	相对浮点精度
	realmax	最大浮点数
	realmin	最小浮点数
	pi	圆周率值 3.141 592 653 589 7……
	i, j	虚数单位
	inf	无穷大
	nan	非数值
	flops	浮点运算次数
	nargin	函数输入变量数

续表

■ 特殊变量和常数		
	nargout	函数输出变量数
	computer	计算机类型
	isieee	当计算机采用 IEEE 算术标准时，其值为真
	why	简明的答案
	version	MATLAB 版本号
■ 时间和日期		
	clock	墙上挂钟
	cputime	CPU 时间(以秒为单位)
	date	日历
	etime	计时函数
	tic	秒表开始执行
	toc	秒表停止
■ 矩阵操作		
	diag	建立或提取对角阵
	fliplr	矩阵作左右翻转
	flipud	矩阵作上下翻转
	reshape	改变矩阵大小
	rot90	矩阵旋转 90°
	tril	提取矩阵的下三角部分
	triu	提取矩阵的上三角部分
	:	矩阵的索引号，重新排列矩阵

表 A.7 特殊矩阵

	compan	友矩阵
	gallery	几个小的测试矩阵
	hadamard	Hadamard 矩阵
	hankel	Hankel 矩阵
	hilb	Hilbert 矩阵
	invhilb	逆 Hilbert 矩阵
	kron	Kronecker 张量积
	magic	魔方矩阵
	pascal	Pascal 矩阵

续表

	rosser	经典的对称特征值测试问题
	toeplitz	Toeplitz 矩阵
	vander	Vandermonde 矩阵
	wilkinson	Wilkinson 特征值测试矩阵

表 A.8　矩阵函数——数值线性代数

■ 矩阵分析		
	cond	计算矩阵条件数
	norm	计算矩阵或向量范数
	rcond	Linpack 逆条件值估计
	rank	计算矩阵秩
	det	计算矩阵行列式值
	trace	计算矩阵的迹
	null	零矩阵
	orth	正交化
	rref	减缩行格式矩阵
■ 线性方程		
	\ 和 /	线性方程求解
	chol	Cholesky 分解
	lu	高斯消元法求系数阵
	inv	矩阵求逆
	qr	正交三角矩阵分解(简称 QR 分解)
	qrdelete	从 QR 分解中消去一列
	qrinsert	在 QR 分解中插入一列
	nnls	非负最小二乘
	pinv	矩阵伪逆
	lscov	协方差已知的情况下最小二乘求解
■ 特征值和奇异值		
	eig	求特征值和特征向量
	poly	求特征多项式
	polyeig	多项式特征值问题
	hess	Hessberg 形式

续表

■ 特征值和奇异值		
	qz	广义特征值
	rsf2csf	变实分块对角阵为复对角形式
	cdf2rdf	变复对角矩阵为实分块对角形式
	schur	Schur 分解
	balance	矩阵均衡处理以提高特征值精度
	svd	奇异值分解
■ 矩阵函数		
	expm	矩阵指数
	expm1	实现 expm 的 M 文件
	expm2	通过泰勒级数求矩阵指数
	expm3	通过特征值和特征向量求矩阵指数
	logm	矩阵对数
	sqrtm	矩阵开平方根
	funm	一般矩阵的计算

表 A.9 稀疏矩阵函数

■ 基本稀疏矩阵		
	speye	稀疏单位矩阵
	sprandn	稀疏随机矩阵
	sprandsym	对称的稀疏随机矩阵
	spdiags	从对角阵中形成稀疏矩阵
■ 完全矩阵和稀疏矩阵之间变换		
	sparse	从非零元素及其序号中形成稀疏矩阵
	full	变稀疏矩阵为完全矩阵
	find	找出非零元素的序号
	spconvert	稀疏矩阵外部结构的变换
■ 稀疏矩阵非零元素的处理		
	nnz	非零元素的数目
	nonzeros	非零元素
	nzmax	分配给非零元素的存储量
	spones	用“1”取代非零元素
	spalloc	为非零元素分配内存

续表

■ 稀疏矩阵非零元素的处理		
	issparse	当矩阵为稀疏矩阵时，其值为真
	spfun	只对非零元素取函数
■ 显示稀疏矩阵		
	spy	显示稀疏结构
	gplot	绘图
■ 排序算法		
	colmmd	列最小度
	symmmd	最小对称度
	symrcm	逆 Cathill - Mckee 序
	colperm	基于非零元素按列排序
	randperm	随机排列向量
	dmperm	Dulmage - Mendelsohn 分解
■ 范数、条件数和秩		
	normest	2 范数估计
	condest	1 范数条件估计
	sprank	结构化秩
■ 树型操作		
	treelayout	显示一个或多个结构树
	treeplot	画结构树
	etree	求矩阵的消元树
	etreeplot	画消元树图
■ 其它		
	symbfact	符号分解分析
	spparms	为稀疏矩阵处理过程设置参数
	spaugment	形成最小二乘增广系统

表 A.10 数据分析和傅里叶变换函数

■ 基本操作		
	max	取最大分量
	min	取最小分量
	mean	求均值
	median	求中值

续表

■ 基本操作		
	std	求标准差
	sort	按升序排列
	sum	求各元素之和
	prod	求各元素之积
	cumsum	求元素累积和
	cumprod	求元素累积积
	trapz	利用梯形法计算数值积分
■ 有限差分		
	diff	计算差分和近似微分
	gradient	计算近似梯度
	del2	5 点离散拉普拉斯变换
■ 向量操作		
	cross	向量的矢量积
	dot	向量的点积
■ 相关		
	corrcoef	求相关系数
	cov	求协方差矩阵
	subspace	子空间之间的夹角
■ 滤波和卷积		
	filter	一维数字滤波器
	filter2	二维数字滤波器
	conv	卷积和多项式乘法
	conv2	二维卷积
	deconv	反卷积和多项式除法
■ 傅里叶变换		
	fft	离散傅里叶变换
	fft2	二维离散傅里叶变换
	ifft	离散逆傅里叶变换
	ifft2	二维离散逆傅里叶变换
	abs	取模(绝对值)
	angle	取相角
	unwrap	删除跨越 360°边界的相角
	fftshift	将零点平移到频谱中心

续表

■ 傅里叶变换		
	cplxpair	将数值分类成复共轭对
	nextpow2	最靠近 2 的幂次

表 A.11　泛函——非线性数值方法

	ode23	低阶法求解常微分方程
	ode23p	低阶法求解常微分方程并绘出结果图形
	ode45	高阶法求解常微分方程
	quad	低阶法计算数值积分
	quad8	高阶法计算数值积分
	fmin	单变量函数的极小化
	fmins	多变量函数的极小化
	fzero	找出单变量函数的零点
	fplot	函数绘图

表 A.12　多项式和内插函数

■ 多项式		
	roots	求多项式根
	poly	构造具有指定根的多项式
	polyval	多项式计算
	polyvalm	带矩阵变量的多项式计算
	residue	部分分式展开(留数计算)
	polyfit	数据的多项式拟合
	polyder	微分多项式
	conv	多项式乘法
	deconv	多项式除法
■ 数据内插		
	interp1	一维数据内插(一维查表)
	interp2	二维数据内插(二维查表)
	interpft	利用 FFT 进行一维数据内插
	griddata	数据网格

续表

■ 样条内插		
	spline	3 次样条数据内插
	ppval	分段多项式计算

表 A.13 通用图形函数

■ 建立和控制图形窗口		
	figure	建立图形(图形窗口)
	gcf	获取当前图形的句柄
	clf	消除当前图形
	close	关闭图形
■ 建立和控制坐标系		
	subplot	在标定位置上建立坐标系
	axes	在任意位置上建立坐标系
	gca	获取当前坐标系的句柄
	cla	消除当前坐标系
	axis	控制坐标系的刻度和形式
	caxis	控制伪彩色坐标刻度
	hold	保持当前图形
■ 句柄图形对象		
	figure	建立图形窗口
	axes	建立坐标系
	line	建立曲线
	text	建立文本串
	patch	建立图形填充块
	surface	建立曲面
	image	建立图像
	uicontrol	建立用户界面控制
	uimenu	建立用户界面菜单
■ 句柄图形操作		
	set	设置对象特性
	get	获取对象特性
	reset	重置对象特性
	delete	删除对象

续表

■ 句柄图形操作		
	gco	获取当前对象的句柄
	drawnow	填充未完成绘图事件
	newplot	预测 NextPlot 性质的 M 文件
	findobj	寻找指定特性值的对象
■ 打印和存储		
	print	打印图形或保存图形
	printopt	配置本地打印机缺省值
	orient	设置纸张取向
	capture	屏幕抓取当前图形
■ 动画		
	moviein	初始化动画帧内存
	getframe	获取动画帧
	movie	播放所记录的动画帧
■ 其它		
	ginput	用鼠标输入图形
	ishold	返回 Hold 状态
	graymon	设置灰度显示器的图形缺省值
	rbbox	涂抹块
	rotate	沿指定方向旋转对象
	terminal	设置图形终端类型
	uiputfile	弹出保存文件的对话框
	uigetfile	弹出询问文件名的对话框
	whitebg	设置白色背景的图形窗口缺省值
	zoom	二维图形的放大、缩小
	waitforbuttonpress	在图形中等待按键/按钮
■ 用户界面工具		
	dialog	主对话框建立 M 文件
	figflag	当图形为当前显示时其值为真
	layout	定义对话框布局参数
	uiguide	关于用户界面约定/标准/建议的说明
■ 对话框		
	errordlg	建立出错对话框
	helpdlg	建立帮助对话框

续表

■ 对话框		
	questdlg	建立提问对话框
	warndlg	建立警告对话框
■ 打印实用工具		
	prtps	PostScript 打印机驱动程序
	prtwin	MS Windows 驱动程序

表 A.14 二维图形函数

■ 基本 X-Y 图形		
	plot	线性图形
	loglog	对数坐标图形
	semilogx	半对数坐标图形(X 轴为对数坐标)
	semilogy	半对数坐标图形(Y 轴为对数坐标)
	fill	绘制二维多边形填充图
■ 特殊 X-Y 图形		
	polar	极坐标图
	bar	条形图
	stem	离散序列图或杆图
	stairs	阶梯图
	errorbar	误差条图
	hist	直方图
	rose	角度直方图
	compass	区域图
	feather	箭头图
	fplot	绘图函数
	comet	星点图
■ 图形注释		
	title	图形标题
	xlabel	X 轴标记
	ylabel	Y 轴标记
	text	文本注释
	gtext	用鼠标放置文本
	grid	网格线

表 A.15　三维图形函数

■ 曲线和区域填充命令		
	plot3	在三维空间中绘制曲线和点
	fill3	在三维空间中绘制并填充三维多边形
	comet3	三维星点图
■ 三维数据的等高线和其它二维图形		
	contour	等高线图
	contour3	三维等高线图
	clabel	在等高线图上标注高度
	contourc	等高线图计算
	pcolor	伪彩色图
	quiver	箭头图
■ 曲面和网格图形		
	mesh	三维网格曲面
	meshc	网格和等高线混合图形
	meshz	带零平面的三维网格图
	surf	三维曲面阴影图
	surfc	曲面和等高线混合图形
	surfl	带亮度的三维曲面阴影图
	waterfall	落差图
■ 立体可视化		
	slice	立体可视图
■ 图形外形		
	view	指定三维图形视点
	viewmtx	显示变换矩阵
	hidden	设置网格消隐方式
	shading	彩色阴影方式
	axis	坐标轴刻度和外形
	caxis	伪彩色坐标轴刻度
	colormap	颜色对照表
■ 图形注释		
	title	图形标题

续表

■ 图形注释		
	xlable	X 轴标记
	ylable	Y 轴标记
	zlable	Z 轴标记
	text	文本注释
	gtext	用鼠标放置文本
	grid	网格线
■ 三维对象		
	cylinder	产生圆柱体
	sphere	产生球

表 A.16　语言结构和调试

■ MATLAB 编程语言		
	script	有关 MATLAB 的底稿文件和 M 文件的说明
	function	增加新的函数
	eval	执行由 MATLAB 表达式构成的字串
	feval	执行由字串指定的函数
	global	定义全局变量
	nargchk	有效的输入变量数
	lasterr	保持出错信息
■ 程序控制流		
	if	条件执行语句
	else	与 if 命令配合使用
	elseif	与 if 命令配合使用
	end	for，while 和 if 语句的结束
	for	重复执行指定次数(循环)
	while	重复执行不定次数(循环)
	break	终止循环的执行
	return	返回引用的函数
	error	显示信息并终止函数执行
■ 交互输入		
	input	提示用户输入
	keyboard	像底稿文件一样使用键盘输入

续表

■ 交互输入		
	menu	产生由用户输入选择的菜单
	pause	等待用户响应
	uimenu	建立用户界面菜单
	uicontrol	建立用户界面控制
■ 调试命令		
	dbstop	设置断点
	dbclear	删除断点
	dbcont	继续执行
	dbdown	改变局部工作空间的内容
	dbstack	列出调用者
	dbstatus	列出所有断点
	dbstep	执行一条或多条语句
	dbtype	带行号列出 M 文件
	dbup	改变局部工作空间的内容
	dbquit	退出调试状态
	mexdebug	调试 MEX 文件

表 A.17　颜色控制和亮度模型函数

■ 颜色控制		
	colormap	颜色对照表
	caxis	伪彩色坐标轴刻度
	shading	彩色阴影方式
■ 颜色板		
	hsv	色彩——饱和度颜色板
	gray	线性灰度颜色板
	hot	黑—红—黄—白颜色板
	cool	深蓝深红阴影颜色板
	bone	以蓝色为基调的灰度颜色板
	copper	线性青铜色调的颜色板
	pink	线性粉红阴影颜色板
	prism	光谱颜色板
	jet	hsv 颜色板的变型

续表

■ 颜色板		
	flag	红、白、蓝、黑交替的颜色板
■ 颜色板相关函数		
	colorbar	显示颜色条
	hsv2rgb	变 HSV 为 RGB 3 色
	rgb2hsv	变 RGB 为 HSV
	contrast	变灰度颜色板为增强图像对比
	brighten	颜色板加亮或变暗
	spinmap	颜色板旋转
	rgbplot	颜色板绘图
■ 加亮模式		
	surfl	带亮度的三维曲面阴影图
	specular	镜面反射
	diffuse	漫反射
	surfnorm	曲面法线

表 A.18 字符串函数

■ 一般函数		
	strings	MATLAB 中有关字符串函数的说明
	abs	变字符串为数值
	setstr	变数值为字符串
	isstr	当变量为字符串时其值为真
	blanks	空串
	deblank	删除尾部的空串
	str2mat	从各个字符串中形成文本矩阵
	eval	执行由 MATLAB 表达式组成的串
■ 字符串比较		
	strcmp	比较字符串
	findstr	在一字符串中查找另一子串
	upper	变字符串为大写
	lower	变字符串为小写
	isletter	当变量为字母时，其值为真
	isspace	当变量为空白字符时，其值为真

续表

■ 字符串比较		
	strrep	取代字符串
	strtok	在字符串中查找标记
■ 字符串与数值之间变换		
	num2str	变数值为字符串
	int2str	变整数为字符串
	str2num	变字符串为数值
	sprintf	变数值为格式控制下的字符串
	sscanf	变字符串为格式控制下的数值
■ 十进制数与十六进制数之间变换		
	hex2num	变十六进制数为 IEEE 标准下的浮点数
	hex2dec	变十六进制数为十进制数
	dec2hex	变十进制数为十六进制数

表 A.19　音频处理函数

■ 一般音频函数		
	sound	变向量为音频信号
	saxis	音频轴刻度
■ 特定计算机音频函数		
	auwrite	写按 Wu－law 编码的音频文件
	auread	读按 Wu－law 编码的音频文件
	wavwrite	写 MS Windows 的.WAV 音频文件
	wavread	读 MS Windows 的.WAV 音频文件
	mu2lin	变 Wu－law 编码音频信号为线性音频信号
	lin2mu	变线性音频信号为 Wu－law 编码音频信号

表 A.20　低级文件 I/O 函数

■ 打开和关闭文件		
	fopen	打开文件
	fclose	关闭文件
■ 未格式化 I/O		
	fread	从文件读二进制数据

续表

■ 未格式化 I/O		
	fwrite	二进制数据写入到文件
■ 格式化 I/O		
	fscanf	从文件中读格式化的数据
	fprintf	将格式化数据写入到文件
	fgetl	从文件中读行，并丢弃换行符
	fgets	从文件中读行，并保持换行符
■ 文件定位		
	ferror	查询文件 I/O 出错状态
	feof	测试文件尾
	fseek	设置文件位置指针
	ftell	获取文件位置指针
	frewind	反绕文件
■ 字符串变换		
	sprintf	将格式化数据写到字符串
	sscanf	从格式化字符串中读取
■ 文件 I/O		
	wk1const	WK1 记录定义
	wk1read	读 WK1 文件
	wk1write	在 WK1 格式化文件中写矩阵
	wk1wrec	写 WK1 记录头
	csvread	从逗号间隔的格式化文件中读一矩阵
	csvwrite	写一矩阵到逗号间隔的格式化文件中
	dlmread	从以 ASCII 码限界的文件中读一矩阵
	dlmwrite	按 ASCII 码限界的文件格式写一矩阵

附录 B

Toolbox 函数

MATLAB 系统提供了许多 Toolbox(工具箱)，而且由于 MATLAB 的可扩充性，Toolbox 的数目与日俱增，这里仅列出一些基本的工具箱函数，以备用户查阅。

表 B.1 为本附录所列出的 Toolbox。表 B.2～B.11 为各个 Toolbox 所提供的工具函数。

表 B.1　MATLAB 提供的部分 Toolbox

目　录　名	工具箱名称	索　　引
local	局部函数库	表 B.2
signal	信号处理	表 B.3
image	图像处理	表 B.4
control	控制系统	表 B.5
ncd	非线性控制设计	表 B.6
robust	鲁棒控制	表 B.7
ident	系统辨识	表 B.8
optim	最优化	表 B.9
nnet	神经网络	表 B.10
fuzzy	模糊系统	表 B.11

表 B.2　局部函数库

	matlabrc	MATLAB 的主启动 M 文件
	printopt	设置打印选项

表 B.3　信号处理工具箱

■ 波形产生		
	sawtooth	产生锯齿波或三角波
	square	产生方波
	sinc	产生 sinc 或$\frac{\sin(\pi t)}{\pi t}$函数
	diric	产生 Dirichlet 或周期 sinc 函数

续表

■ 滤波器分析和实现		
	abs	取绝对值(幅值)
	angle	取相角
	conv	求卷积
	fltfilt	重叠相加法 FFT 滤波器实现
	filter	直接滤波器实现
	filtfilt	零相位数字滤波
	filtic	filter 函数初始条件选择
	freqs	模拟滤波器频率响应
	freqspace	频率响应中的频率间隔
	freqz	数字滤波器频率响应
	grpdelay	平均滤波延迟(群延迟)
	impz	数字滤波器的冲激响应
	zplane	离散系统零极点图
■ 线性系统变换		
	convmtx	卷积矩阵
	poly2rc	从多项式系数中计算反射系数
	rc2poly	从反射系数中计算多项式系数
	residuez	Z 变换部分分式展开或留数计算
	sos2ss	变系统二阶分割形式为状态空间形式
	sos2tf	变系统二阶分割形式为传递函数形式
	sos2zp	变系统二阶分割形式为零极点增益形式
	ss2sos	变系统状态空间形式为二阶分割形式
	ss2tf	变系统状态空间形式为传递函数形式
	ss2zp	变系统状态空间形式为零极点增益形式
	tf2ss	变系统传递函数形式为状态空间形式
	tf2zp	变系统传递函数形式为零极点增益形式
	zp2sos	变系统零极点增益形式为二阶分割形式
	zp2ss	变系统零极点增益形式为状态空间形式
	zp2tf	变系统零极点增益形式为传递函数形式

续表

■ IIR 滤波器设计		
	besself	Bessel(贝塞尔)模拟滤波器设计
	butter	Butterworth(比特沃思)滤波器设计
	cheby1	Chebyshev(切比雪夫)Ⅰ型滤波器设计
	cheby2	Chebyshev(切比雪夫)Ⅱ型滤波器设计
	ellip	椭圆滤波器设计
	yulewalk	递归数字滤波器设计
■ IIR 滤波器阶的选择		
	buttord	Butterworth 滤波器阶的选择
	cheb1ord	Chebyshev Ⅰ型滤波器阶的选择
	cheb2ord	Chebyshev Ⅱ型滤波器阶的选择
	ellipord	椭圆滤波器阶的选择
■ FIR 滤波器设计		
	fir1	基于窗函数的 FIR 滤波器设计——标准响应
	fir2	基于窗函数的 FIR 滤波器设计——任意响应
	firls	最小二乘 FIR 滤波器设计
	intfilt	内插 FIR 滤波器设计
	remez	Parks - McCellan 最优 FIR 滤波器设计
	remezord	Parks - McCellan 最优 FIR 滤波器阶估计
■ 变换		
	czt	线性调频 Z 变换
	dct	离散余弦变换(DCT)
	idct	逆离散余弦变换
	dftmtx	离散傅里叶变换矩阵
	fft	一维快速傅里叶变换
	ifft	一维逆快速傅里叶变换
	fftshift	重新排列 FFT 的输出
	hilbert	Hilbert(希尔伯特)变换
■ 统计信号处理		
	cov	协方差矩阵
	xcov	互协方差函数估计
	corrcoef	相关系数矩阵

续表

■ 统计信号处理		
	xcorr	互相关函数估计
	cohere	相关函数平方幅值估计
	csd	互谱密度(CSD)估计
	psd	信号功率谱密度(PSD)估计
	tfe	从输入输出中估计传递函数
■ 窗函数		
	boxcar	矩形窗
	triang	三角窗
	bartlett	Bartlett(巴特利特)窗
	hamming	Hamming(哈明)窗
	hanning	Hanning(汉宁)窗
	blackman	Blackman(布莱克曼)窗
	chebwin	Chebyshev 窗
	kaiser	Kaiser 窗
■ 参数化建模		
	invfreqs	模拟滤波器拟合频率响应
	invfreqz	离散滤波器拟合频率响应
	prony	利用 Prony 方法的离散滤波器拟合时间响应
	stmcb	利用 Steiglitz - McBride 迭代方法求线性模型
	levinson	Levinson - Durbin 递归算法
	lpc	线性预测系数
■ 特殊操作		
	rceps	实倒谱和最小相位重构
	cceps	倒谱分析和最小相位重构
	decimate	降低序列的取样速率
	interp	提高取样速率(内插)
	resample	改变取样速率
	medfilt1	一维中值滤波
	deconv	反卷积和多项式除法
	modulate	通讯仿真中的调制
	demod	通讯仿真中的解调

续表

■ **特殊操作**		
	vco	电压控制振荡器
	specgram	频谱分析
■ **模拟原型滤波器设计**		
	besselap	Bessel 模拟低通滤波器原型
	buttap	Butterworth 模拟低通滤波器原型
	cheb1ap	Chebyshev Ⅰ 型模拟低通滤波器原型
	cheb2ap	Chebyshev Ⅱ 型模拟低通滤波器原型
	ellipap	椭圆模拟低通滤波器原型
■ **频率变换**		
	lp2bp	低通到带通模拟滤波器变换
	lp2hp	低通到高通模拟滤波器变换
	lp2bs	低通到带阻模拟滤波器变换
	lp2lp	低通到低通模拟滤波器变换
■ **滤波器离散化**		
	bilinear	双线性变换
	impinvar	冲激响应不变法实现模拟到数字的滤波器变换
■ **其它**		
	conv2	二维卷积
	cplxpair	将复数归成复共轭对
	detrend	删除线性趋势
	fft2	二维快速傅里叶变换
	ifft2	二维逆快速傅里叶变换
	filter2	二维数字滤波器
	polystab	稳定多项式
	xcorr2	二维互相关

表 B.4 图像处理工具箱

■ 图像输入/输出		
	bmpread	从磁盘中读 BMP(Microsoft Windows 下的位图)文件
	bmpwrite	将一 BMP 文件写入到磁盘
	gifread	从磁盘中读 GIF 文件
	gifwrite	将 GIF 文件写入磁盘
	hdfpeek	在 HDF 文件中列出目标标记/参考对
	hdfread	从 HDF 文件中读取数据
	hdwrite	写数据到 HDF 文件中
	pcxread	从磁盘中读 PCX 文件
	pcxwrite	将 PCX 文件写入磁盘
	tiffread	从磁盘中读 TIFF 文件
	tiffwrite	将 TIFF 文件写入磁盘
	xwdread	从磁盘中读 XWD 文件
	xwdwrite	将 XWD 文件写入磁盘
■ 实用程序		
	getimage	从坐标系中读取图像数据
	isbw	当图像为黑白图像时，其值为真
	isgray	当图像为灰度图像时，其值为真
	isind	当图像为加标图像时，其值为真
■ 颜色操作		
	brighten	加亮或增暗一颜色板
	cmunique	寻找唯一的颜色板及相应的图像
	cmpermute	置换颜色板位置
	cmgamma	γ 校正颜色板
	cmgamdef	缺省的 γ 校正表
	dither	Floyd－Steinberg 图像颤抖算法
	hsv2rgb	变 HSV 值为 RGB 颜色空间
	imadjust	调整并增强图像强度
	imapprox	利用更少颜色的图像逼近加标图像

续表

■ 颜色操作		
	ntsc2rgb	变 NTSC 值为 RGB 颜色空间
	rgb2gray	变 RGB 值为灰度值
	rgb2hsv	变 RGB 值为 HSV 颜色空间
	rgb2ntsc	变 RGB 值为 NTSC 颜色空间
	rgbplot	绘制 RGB 颜色板分量的图形
■ 几何操作		
	imcrop	修剪图像
	imresize	改变图像大小
	imrotate	旋转图像
	truesize	改变图像大小使之具有实际尺寸
	imzoom	放大或缩小图像和二维图形
■ 图像增强/分析		
	brighten	增强成削弱颜色板
	grayslice	密度(强度)限幅
	histeq	直方图均衡化
	imadjust	调整和展宽图像强度
	imapprox	利用较少颜色的图像逼近图像
	imhist	图像直方图
	impixel	一像素点的颜色
	improfile	轮廓强度
	interp2	二维数据内插
■ 图像统计		
	mean2	矩阵的均值
	corr2	二维相关系数
	std2	二维标准差
■ 形态操作		
	bwarea	二进制图像中的目标区域
	dilate	加浓二进制图像
	erode	冲淡二进制图像
	edge	边界提取
	bweuler	欧拉数
	bwmorph	形态算子
	bwperim	二进制图像中目标的周围

续表

■ FIR(有限冲激响应)滤波器设计		
	fsamp2	通过频率取样的二维 FIR 滤波器设计
	fspecial	特殊的二维滤波器
	ftrans2	通过频率变换的二维 FIR 滤波器设计
	fwind1	使用一维窗函数的 FIR 滤波器设计
	fwind2	使用二维窗函数的 FIR 滤波器设计
	imnoise	图像噪声
■ 频率响应		
	freqspace	二维频率响应的频率空间
	freqz2	二维频率响应
■ 滤波		
	colfilt	局部非线性滤波
	conv2	二维卷积
	filter2	二维滤波
	medfilt2	二维中值滤波
	mfilter2	屏蔽滤波
	nlfilter	局部非线性滤波
	wiener2	自适应二维维纳滤波
■ 分块处理		
	bestblk	分块处理的最佳块大小
	blkproc	按块处理一图像
	col2im	重新排列以形成图像
	colfilt	局部非线性滤波
	im2col	重新排列成列
■ 个别区域		
	mfilter2	屏蔽滤波
	roipoly	定义感兴趣的多边区域
	roicolor	用颜色定义感兴趣的区域
■ 变换		
	dct2	二维离散余弦变换
	fft2	二维快速傅里叶变换
	fftshift	零频移到频谱中心
	idct2	二维逆离散余弦变换

续表

■ **变换**		
	ifft2	二维逆快速傅里叶变换
	radon	Radon 变换
■ **转换**		
	dither	Floyd - Steinberg 图像颤抖
	gray2ind	变灰度图像为附标图像
	hsv2rgb	变 HSV 值为 RGB 值
	im2bw	变图像为黑白图形
	imslice	在图像中获取/置入图像块
	ind2gray	变附标图像为灰度图像
	ind2rgb	变附标图像为 RGB 图像
	mat2gray	变矩阵为(灰度)图像
	ntsc2rgb	变 NTSC 值为 RGB 值
	rgb2gray	变 RGB 图像或值为灰度图像或值
	rgb2hsv	变 RGB 值为 HSV 值
	rgb2ind	变 RGB 图像为附标图像
	rgb2ntsc	变 RGB 值为 NTSC 值
■ **图像显示**		
	colorbar	显示颜色条
	colormap	设置或获取颜色查找表
	gray	线性灰度颜色板
	hsv, hot, jet	颜色板
	image	显示附标图像
	imagesc	数据定标并按图像显示
	imcontour	图像等高线
	immovie	制作图像动画
	imshow	显示所有类型的图像数据
	montage	按矩形剪辑方式显示图像
	subimage	显示多个图像
	warp	将图像卷成曲面
■ **演示**		
	imdemo	一般图像处理演示
	dctdemo	二维离散余弦变换图像压缩演示
	firdemo	二维 FIR 滤波器演示

续表

■ 演示		
	nlfdemo	二维非线性滤波演示
■ 专用函数		
	cumsum3d	三维矩阵封装成二维矩阵时的累积和
	dct	一维离散余弦变换
	dctmtx2	一元二维离散余弦变换矩阵
	ditherc	图像颤抖的 MEX 文件
	elem3d	三维矩阵封装成二维矩阵的元素位置
	getline	利用橡皮线跟踪鼠标移动
	getpts	利用可视点跟踪鼠标移动
	getrect	利用橡皮矩形跟踪鼠标移动
	gif	压缩 GIF 数据
	hdfreadc	读 HDF 文件的 MEX 文件
	hdfpeekc	搜索 HDF 文件的 MEX 文件
	hdfwc	写 HDF 文件的 MEX 文件
	idct	一维逆离散余弦变换
	im2gray	变图像为灰度
	imhistc	图像直方图计算的 MEX 文件
	ndx3d	三维矩阵封装成二维矩阵的索引
	rgb2im	变 RGB 图像为附标或强度图像
	rle	压缩编码数据
	tiff	压缩 tiff 编码数据
	vmquant	与彩色量化 MEX 文件接口的 M 文件
	waitbar	显示等待条
■ MAT 文件		
	bwmorph. mat	bwmorph. m 文件的查找表
	forest. mat	Carmanah Old Growth Forest 的扫描相片
	mri. mat	人体心脑的磁性共振图像
	trees. mat	树的扫描图像

表 B.5 控制系统工具箱

■ 建模		
	append	追加系统动态特性
	augstate	变量状态作为输出
	blkbuild	从方框图中构造状态空间系统
	cloop	系统的闭环
	connect	方框图建模
	conv	两个多项式的卷积
	destim	从增益矩阵中形成离散状态估计器
	dreg	从增益矩阵中形成离散控制器和估计器
	drmodel	产生随机离散模型
	estim	从增益矩阵中形成连续状态估计器
	feedback	反馈系统连接
	ord2	产生二阶系统的 A、B、C、D
	pade	时延的 Pade 近似
	parallel	并行系统连接
	reg	从增益矩阵中形成连续控制器和估计器
	rmodel	产生随机连续模型
	series	串行系统连接
	ssdelete	从模型中删除输入、输出或状态
	ssselect	从大系统中选择子系统
■ 模型变换		
	c2d	变连续系统为离散系统
	c2dm	利用指定方法变连续为离散系统
	c2dt	带一延时变连续为离散系统
	d2c	变离散为连续系统
	d2cm	利用指定方法变离散为连续系统
	poly	变根值表示为多项式表示
	residue	部分分式展开
	ss2tf	变状态空间表示为传递函数表示
	ss2zp	变状态空间表示为零极点表示
	tf2ss	变传递函数表示为状态空间表示
	tf2zp	变传递函数表示为零极点表示
	zp2tf	变零极点表示为传递函数表示
	zp2ss	变零极点表示为状态空间表示

续表

■ 模型简化		
	balreal	平衡实现
	dbalreal	离散平衡实现
	dmodred	离散模型降阶
	minreal	最小实现和零极点对消
	modred	模型降阶
■ 模型实现		
	canon	正则形式
	ctrbf	可控阶梯形
	obsvf	可观阶梯形
	ss2ss	采用相似变换
■ 模型特性		
	covar	相对于白噪声的连续协方差响应
	ctrb	可控性矩阵
	damp	阻尼系数和固有频率
	dcgain	连续稳态(直流)增益
	dcovar	相对于白噪声的离散协方差响应
	ddamp	离散阻尼系数和固有频率
	ddcgain	离散稳态(直流)增益
	dgram	离散可控性和可观性
	dsort	按幅值排序离散特征值
	eig	特征值和特征向量
	esort	按实部排序连续特征值
	gram	可控性和可观性
	obsv	可观性矩阵
	printsys	按格式显示系统
	roots	多项式之根
	tzero	传递零点
	tzero2	利用随机扰动法传递零点
■ 时域响应		
	dimpulse	离散时间单位冲激响应
	dinitial	离散时间零输入响应
	dlsim	任意输入下的离散时间仿真
	dstep	离散时间阶跃响应

续表

■ 时域响应		
	filter	单输入单输出 Z 变换仿真
	impulse	冲激响应
	initial	连续时间零输入响应
	lsim	任意输入下的连续时间仿真
	ltitr	低级时间响应函数
	step	阶跃响应
	stepfun	阶跃函数
■ 频域响应		
	bode	Bode(波特)图(频域响应)
	dbode	离散 Bode 图
	dnichols	离散 Nichols 图
	dnyquist	离散 Nyquist 图
	dsigma	离散奇异值频域图
	fbode	连续系统的快速 Bode 图
	freqs	拉普拉斯变换频率响应
	freqz	Z 变换频率响应
	ltifr	低级频率响应函数
	margin	增益和相位裕度
	nichols	Nichols 图
	ngrid	画 Nichols 图的栅格线
	nyquist	Nyquist 图
	sigma	奇异值频域图
■ 根轨迹		
	pzmap	零极点图
	rlocfind	交互式地确定根轨迹增益
	rlocus	画根轨迹
	sgrid	在 ω_n, z 网格上画连续根轨迹
	zgrid	在 ω_n, z 网格上画离散根轨迹
■ 增益选择		
	acker	单输入单输出极点配置
	dlqe	离散线性二次估计器设计
	dlqew	离散线性二次估计器设计
	dlqr	离散线性二次调节器设计

续表

■ **增益选择**		
	dlqry	输出加权的离散调节器设计
	lqe	线性二次估计器设计
	lqed	基于连续代价函数的离散估计器设计
	lqe2	利用 Schur 法设计线性二次估计器
	lqew	一般线性二次估计器设计
	lqr	线性二次调节器设计
	lqrd	基于连续代价函数的离散调节器设计
	lqry	输出加权的调节器设计
	lqr2	利用 Schur 法设计线性二次调节器
	place	极点配置
■ **方程求解**		
	are	代数 Riccati 方程求解
	dlyap	离散 Lyapunov 方程求解
	lyap	连续 Lyapunov 方程求解
	lyap2	利用对角化求解 Lyapunov 方程
■ **演示示例**		
	ctrldemo	控制工具箱介绍
	boildemo	锅炉系统的 LQG 设计
	jetdemo	喷气式飞机偏航阻尼的典型设计
	diskdemo	硬盘控制器的数字控制
	kalmdemo	Kalman 滤波器设计和仿真
■ **实用工具**		
	abcdchk	检测(A，B，C，D)组的一致性
	chop	取 n 个重要的位置
	dexresp	离散取样响应函数
	dfrqint	离散 Bode 图的自动定范围的算法
	dfrqint2	离散 Nyquist 图的自动定范围的算法
	dmulresp	离散多变量响应函数
	dists1	到直线间的距离
	dric	离散 Riccati 方程留数计算
	dsigma2	DSIGMA 实用工具函数
	dtimvec	离散时间响应的自动定范围算法
	exresp	取样响应函数

续表

■ 实用工具		
	freqint	Bode 图的自动定范围算法
	freqint2	Nyquist 图的自动定范围算法
	freqresp	低级频率响应函数
	givens	旋转
	housh	构造 Householder 变换
	imargin	利用内插技术求增益和相位裕度
	lab2ser	变标号为字符串
	mulresp	多变量响应函数
	nargchk	检测 M 文件的变量数
	perpxy	寻找最近的正交点
	poly2str	变多项式为字符串
	printmat	带行列号打印矩阵
	ric	Riccati 方程留数计算
	schord	有序 Schwr 分解
	sigma2	SIGMA 实用工具函数
	tfchk	检测传递函数的一致性
	timvec	连续时间响应的自动定范围算法
	tzreduce	在计算过零点时简化系统
	vsort	匹配两根轨迹的向量

表 B.6　非线性控制设计工具箱

■ 对话框管理		
	coneddlg	管理 NCD 工具箱固定编辑器的对话框
	paramdlg	管理 NCD 优化参数的对话框
	rangedlg	管理坐标系范围的对话框
	refdlg	管理 NCD 参考信号的对话框
	stepdlg	管理 NCD 阶跃响应的对话框
	uncerdlg	管理 NCD 不确定变量的对话框
■ 主要界面		
	contrncd	建立 NCD 固定图形的用户界面控制
	menuncd	建立 NCD 固定图形的用户界面菜单
	ncdblock	包含 NCD 框图的 SIMULINK 系统
	optblock	打开一个 NCD 图形的底稿文件
	optfig	建立一个 NCD 固定图形

续表

■ 主要优化		
	costfun	NCD 优化的代价函数
	nlinopt	执行优化算法
■ 演示示例		
	ncddemo	包含所有 NCD 演示示例的 SIMULINK 系统
	ncddemo1	PID 控制器
	ncddemo2	带前馈控制器的 LQR
	ncddemo3	多输入多输出的 PI 控制器
	ncddemo4	倒摆演示
■ 教程		
	ncdtut1	控制设计示例
	ncdtut2	系统辨识示例
■ 用户界面工具		
	dialog	主对话框建立 M 文件
	errordlg	建立出错对话框
	figflag	当图形为当前显示在屏幕上时，其值为真
	helpdlg	显示一帮助对话框
	layout	定义对话框布局参数的底稿文件
	questdlg	建立提问对话框
	uiguide	有关用户界面约定/标准/建议的说明
	warndlg	建立警告对话框
■ 演示和教程实用工具		
	ncd1init	为 ncddemo1 的优化进行设置
	ncd2init	为 ncddemo2 的优化进行设置
	ncd3init	为 ncddemo3 的优化进行设置
	ncd4init	为 ncddemo4 的优化进行设置
	penddata	为 ncdtut2(即倒摆)进行设置
■ 界面实用工具		
	curobj	提供有关当前点的信息
	dividecb	将固定界分为两部分
	delline	从 NCD 图中删除所有的图
	donep	收回 Close 按钮和菜单
	errorncd	管理 NCD 产生的常见错误，它调用 errordlg(出错对话框)

续表

■ 界面实用工具		
	fillaxes	建立约束边界并进行数据检测
	forceit	在已存在的界限内插入一子集
	keyncd	NCD 按键函数
	loadncd	装入并显示 NCD 数据
	makesurf	建立并限界曲面
	snapncd	以 22.5°间隔排出约束条
	refresho	使约束矩阵与图形一致
	saveload	当文件是从 SelectFile 中选择时，其值为真
	texted	收回 Port 可编辑的文本
	undoncd	放弃上次 NCD 图形用户界面的操作
	updatdlg	更新 NCD 对话框
■ 最优化实用工具		
	convertm	变约束矩阵为最优化格式
	minipars	NCD 最小化分析
	montevar	初始化 Monte Carlo 仿真
	ncdglob	定义 NCD 全局变量
	str2mat2	变一行字符串为多行字符串
■ 帮助文本文件(以.HLP 为扩展名)		
	hotkey	热键帮助
	mainncd	一般 NCD 帮助
	paramdlg	最优化参数对话框的帮助
	readncd	与 README.M 文件内容相同
	stepdlg	阶跃响应对话框的帮助
	uncerdlg	不确定性变量对话框的帮助

表 B.7 鲁棒控制工具箱

■ 可选系统数据结构		
	branch	从树中提取一分支
	graft	在树中增加一分支
	issystem	辨识一系统变量
	istree	辨识一树型变量

续表

■ 可选系统数据结构		
	mksys	为系统建立树变量
	tree	建立树变量
	vrsys	返回标准系统变量名
■ 建模		
	augss	系统增广(状态空间模型)
	augtf	系统增广(传递函数模型)
	interc	一般多变量内连系统
■ 模型转换		
	bilin	多变量双线性变换
	des2ss	利用奇异值分解变系统为状态空间系统
	lftf	线性分式变换
	sectf	扇形变换
	stabproj	稳定和逆稳定映射
	slowfast	慢/快分解
	tfm2ss	变传递函数模型为状态空间模型
■ 实用工具		
	aresolv	广义连续时间 Riccati 方程求解
	daresolv	广义离散时间 Riccati 方程求解
	riccond	连续时间 Riccati 方程的条件数
	driccond	离散时间 Riccati 方程的条件数
	blkrsch	通过 cschur 得到块有序实 Schur 形式
	cschur	通过复旋转得有序复 Schur 形式
■ 多变量 Bode 图		
	cgloci	连续特性增益轨迹
	dcgloci	离散特性增益轨迹
	dsigma	离散奇异值 Bode 图
	muopt	具有实/复数混合不确定性系统的 SSV(结构化奇异值)上界
	osborne	通过 Osborne 法求得的 SSV 上界
	perron	计算 Perron 特征值
	psv	Perron 特征结构的 SSV
	sigma	连续奇异值 Bode 图
	ssv	结构化奇异值 Bode 图

续表

■ 因子分解技术		
	iofc	内外因子分解(列类型)
	iofr	内外因子分解(行类型)
	sfl	左边频谱分解
	sfr	右边频谱分解
■ 模型简化方法		
	balmr	截断均衡模型简化
	bstschml	相对误差 Schur 模型简化
	bstschmr	相对误差 Schur 模型简化
	imp2ss	从脉冲响应到状态空间实现
	obalreal	有序均衡实现
	ohklmr	最优 Hankel 极小化逼近
	rschur	Schur 模型简化
■ 鲁棒控制综合方法		
	h2lqg	连续时间 H_2 综合
	dh2lqg	离散时间 H_2 综合
	hinf	连续时间 H_∞综合
	dhinf	离散时间 H_∞综合
	hinfopt	H_∞综合的 γ 迭代
	normh2	计算 H_2 范数
	normhinf	计算 H_∞范数
	lqg	LQG 最优控制综合
	ltru	LQG 闭环传递补偿
	ltry	LQG 闭环传递补偿
	youla	Youla 参数化
■ 演示示例		
	accdemo	弹簧质量标准问题
	dintdemo	双积分器系统的 H_∞设计
	hinfdemo	飞机或大型空间结构的 H_2 或 H_∞设计示例
	ltrdemo	LQR/LTR 设计示例：飞机
	mudemo	μ 综合示例
	mudemo1	μ 综合示例
	mrdemo	鲁棒模型简化示例
	rctdemo	鲁棒控制工具箱演示——主菜单

表 B.8 系统辨识工具箱

■ 仿真和预测		
	idsim	仿真一给定的系统
	pe	计算预测误差
	poly2th	从给定的多项式中构造 Θ 矩阵
	predict	M 步超前预测
■ 数据处理		
	dtrend	从数据集中删除方位
	idfilt	通过 Butterworth 滤波器对数据进行滤波
■ 非参数化估计		
	covf	估计数据矩阵的协方差矩阵
	cra	相关分析
	etfe	估计经验传递函数并计算周期图
	spa	频谱分析
■ 参数估计		
	ar	利用各种方法的 AR 信号模型
	armax	ARMAX 模型预测误差估计
	arx	ARX 模型的最小二乘估计
	bj	Box - Jenkins 模型的预测误差估计
	canstart	具有初值参数估计的多变量模型
	ivar	时间序列的 AR 部分的仪器 Ⅳ 估计
	ivx	单输出 ARX 模型的仪器可变估计
	iv4	ARX 模型近似最优的 Ⅳ 估计
	oe	输出误差模型的预测误差估计
	pem	一般线性模型的预测误差估计
■ 建立模型结构		
	arx2th	ARX 模型的 Θ 格式
	canform	正则形模型结构
	mf2th	将用户定义的模型结构封装入 Θ 模型格式中
	modstruc	在 ms2th 函数中使用的模型结构
	ms2th	将标准状态空间参数封装入 Θ 格式中
	poly2th	从给定多项式中产生 Θ 矩阵

续表

■ 处理模型结构		
	fixpar	在状态空间和 ARX 模型结构中，找出要修正的参数
	sett	在 Θ 结构中设置取样间隔
	thinit	参数的(随机)初始值
	unfixpar	在状态空间和 ARX 模型结构中，放松参数
■ 模型变换		
	th2arx	变 Θ 格式模型为 ARX 模型
	th2ff	求模型的频率响应及标准偏差
	th2par	变 Θ 格式为参数和协方差阵
	th2poly	求给定模型相应的多项式
	th2ss	变 Θ 格式为状态空间表示
	th2tf	变 Θ 格式为传递函数表示
	th2zp	求零极点、静态增益和标准偏差
	thc2thd	变连续时间模型为离散时间模型
	thd2thc	变离散时间模型为连续时间模型
■ 模型表示		
	bodeplot	传递函数的 Bode 图或频谱
	ffplot	频域函数
	idplot	输入——输出数据
	nyqplot	传递函数的 Nyquist 图
	present	屏幕上的参数模型
	zpplot	零点和极点
■ 信息提取		
	getmfth	获取定义模型结构的 M 文件的文件名
	getncap	获取数据点数和参数个数
	getff	选取频率函数
	gett	为某模型获取取样间隔
	getzp	在由 th2zp 函数产生的零极点格式中，提取零点和极点
■ 模型合法化		
	compare	将仿真和预测的输出与测量输出比较
	idsim	仿真一给定的系统
	pe	预测误差

续表

■ 模型合法化		
	predict	M 步超前预测
	resid	计算和测试与某模型相关的留数
■ 估计模型的不确定性		
	idsimsd	在仿真模型响应中说明不确定性
	th2ff	模型频率函数和标准偏差
	th2zp	零点、极点、静态增益及其标准偏差
■ 模型结构选择		
	arxstruc	ARX 模型类的损失函数
	ivstruc	单输出类的输出误差拟合
	selstruc	根据各种准则选择模型结构
	struc	arxstruc 和 ivstruc 的典型结构矩阵
■ 递归参数估计		
	rarx	对 AR 模型递归计算估值
	rarmax	对 ARMAX 模型递归计算估值
	rbj	对 Box - Jenkins 模型递归计算估值
	roe	对输出误差模型递归计算估值
	rpem	对一般模型递归计算估值
	rplr	对一般模型递归计算估值
	segment	分段数据并跟踪快变系统

表 B.9 最优化工具箱

■ 非线性最小化函数		
	attgoal	达到多目标
	constr	约束极小化
	fmin	无约束极小化(标量情况)
	fminu	利用梯度搜索的无约束极小化
	fmins	利用单纯形搜索的无约束极小化
	fsolve	非线性方程求解
	leastsq	非线性最小二乘
	minimax	极小极大求解
	seminf	半定极小化

续表

■ 矩阵问题极小化		
	lp	线性规划
	qp	二次规划
	nnls	非负最小二乘
■ 控制缺省值和选项		
	foptions	参数设置
■ 演示		
	optdemo	演示菜单
	tutdemo	启动教程
	bandemo	香蕉型函数的极小化
	goaldemo	目标达到
	dfildemo	有限精度滤波器设计
	datdemo	数据拟合成曲线
■ 内部使用的实用程序		
三次内插程序		
	cubic	内插 4 点以找出极大值
	cubici1	内插 2 点和梯度，以估计极小值
	cubici2	内插 3 点和 1 梯度
	cubici3	内插 2 点和梯度，以找出步长和极小值
二次内插程序		
	quad2	内插 3 点以找出极大值
	quadinter	内插 3 点以估计极小值
演示实用程序		
	eigfun	返回分类特征值的函数
	elimone	消去一变量
	filtfun	频率响应和根
	filtfun2	频率响应范数和根
	fitfun	返回拟合数据中的误差范数
	fitfun2	返回拟合数据中的误差矢量
半定实用程序		
	semifun	半定问题转换成约束问题
	findmax	在数据向量中内插极大值

续表

半定实用程序		
	findmax2	在数据矩阵中内插极大值
	v2sort	分类两向量，然后删去丢失的元素
目标达到的实用程序		
	goalfun	目标达到问题转换成约束条件问题
	goalgra	变换目标达到问题中的梯度
测试程序		
	toptim	最优化测试组
	toptimf	最优化测试组的测试函数
	toptimg	最优化测试组的测试函数梯度
其它		
	graderr	用于检查梯度的不一致性
	lsint	初始化最小二乘程序的函数
	optint	初始化无约束极小化程序的函数
	searchq	线性搜索程序

表 B.10 神经网络工具箱

■ 误差分析函数		
	errsurf	计算误差曲面
	plotep	在误差曲面上绘制权和基位置图
	plotes	绘制误差曲面图
■ δ 函数		
	deltalin	对 PURELIN 神经元的 δ 函数
	deltalog	对 LOGSIG 神经元的 δ 函数
	deltatan	对 TANSIG 神经元的 δ 函数
■ 设计		
	solvehop	设计 Hopfield 网络
	solvelin	设计线性网络
	solverb	设计径向基网络
	solverbe	设计精确的径向基网络

续表

■ 初始化		
	initc	竞争层初始化
	initelm	Elman 递归网络初始化
	initff	至多三层的前向网络初始化
	initlin	线性层初始化
	initlvq	LVQ 网络初始化
	initp	感知层初始化
	initsm	自组织映射初始化
	midpoint	产生中点值
	nwlog	对 LOGSIG 神经元产生 Nguyen - Widrow 随机数
	nwtan	对 TANSIG 神经元产生 Nguyen - widrow 随机数
	randnc	产生归一化列随机数
	randnr	产生归一化行随机数
	rands	产生对称随机数
■ 学习规则		
	learnbp	反向演播学习规则
	learnbpm	带预测的反向演播学习规则
	learnh	Hebb 学习规则
	learnhd	退化的 Hebb 学习规则
	learnis	内星学习规则
	learnk	Kohonen 学习规则
	learnlm	Levenberg - Marquardt 学习规则
	learnlvq	学习矢量量化规则
	learnos	外星学习规则
	learnp	感知层学习规则
	learnpn	归一化的感知层学习规则
	learnwh	Widrow - Hoff 学习规则
■ 矩阵		
	combvec	创建所有的矢量集
	delaysig	从信号矩阵中建立退化的信号矩阵
	dist	计算矢量距离

续表

■ 矩阵		
	ind2vec	变下标矢量为稀疏矩阵表示
	normc	归一化矩阵列
	normr	归一化矩阵行
	pnormc	伪归一化矩阵列
	quant	离散化成某数值的整数倍
	sumsqr	平方和
	vect2ind	变稀疏矩阵表示为下标矢量
■ 邻域		
	nbdist	使用矢量距离的邻域阵
	nbgrid	使用栅格距离的邻域阵
	nbman	使用 Manhattan 距离的邻域阵
■ 绘图		
	barerr	每个输出矢量的误差条形图表
	hintonw	绘制权值图
	hintonwb	绘制权值和偏差图
	ploterr	绘出网络误差与时间的关系
	plotes	绘制误差曲面
	plotfa	绘出目标模式及网络函数的逼近
	plotpv	绘出限幅神经元的感知器分类
	plotsm	绘制自组织映射图
	plottr	绘出网络误差记录及自适应学习速率
	plotvec	用不同颜色绘制矢量
■ 仿真		
	simuc	竞争层仿真
	simuelm	Elman 递归网络仿真
	simuff	前向网络仿真
	simuhop	Hopfield 网络仿真
	simulin	线性层仿真
	simup	感知层仿真
	simurb	径向基网络仿真
	simusm	自组织映射仿真

续表

■ 训练		
	trainbp	利用反向演播训练前向网络
	trairbpx	利用快速反向演播训练网络
	trainc	训练竞争层网络
	trainelm	训练 Elman 递归网络
	trainlvq	训练 LVQ 网络
	trainp	利用感知规则训练感知层
	trainpn	利用归一化感知规则训练感知层
	trainsm	利用 Kohonen 规则训练自组织映射
	trainwh	利用 Widrow Hoff 规则训练线性层
■ 传递函数		
	compet	竞争层传递函数
	hardlim	硬限幅传递函数
	hardlims	对称硬限幅传递函数
	logsig	对数 S 型传递函数
	purelin	线性传递函数
	radbas	径向基传递函数
	satlins	对称饱和线性传递函数
	tansig	正切 S 型传递函数

表 B.11　模糊系统工具箱

■ GUI 编辑器		
	fuzzy	基本 FIS(模糊推理系统)编辑器
	mfedit	隶属度函数编辑器
	ruleedit	规则编辑器及(句法)分析程序
	releview	规则观察器及模糊推理框图
	surfview	输出曲面观测器
■ 隶属度函数		
	dsigmf	两个“S”形隶属度函数的差
	gauss2mf	双边高斯曲线隶属度函数
	gaussmf	高斯曲线隶属度函数
	gbellmf	广义钟形隶属度函数

续表

	pimf	π形隶属度函数
	psigmf	两个"S"形隶属度函数的积
	smf	"S"形隶属度函数
	sigmf	"sigmoid(S)"形隶属度函数
	trapmf	梯形隶属度函数
	trimf	三角形隶属度函数
	zmf	"Z"形隶属度函数
■ 命令行 FIS 函数		
	addmf	将隶属度函数加到 FIS 中
	addrule	将规则加到 FIS 中
	addvar	将变量加到 FIS 中
	defuzz	去模糊隶属度函数
	evalfis	完成模糊推理计算
	evalmf	隶属度函数计算
	gensurf	产生 FIS 输出曲面
	getfis	获得模糊系统的特性
	mf2mf	在函数之间变换参数
	newfis	产生新的 FIS
	parsrule	分析模糊规则
	plotfis	显示 FIS 输入/输出图
	plotmf	显示出一个变量的所有隶属度函数
	readfis	从磁盘中装入 FIS
	rmmf	从 FIS 删除隶属度函数
	rmvar	从 FIS 中删除变量
	setfis	设置模糊系统特性
	showfis	显示带注释的 FIS
	showrule	显示 FIS 规则
	writefis	在磁盘中保存 FIS
■ 先进技术		
	anfis	Sugeno－type FIS 的训练程序
	fcm	利用模糊 C 平均聚集方法找出簇
	genfis1	利用一般方法产生 FIS 矩阵
	genfis2	利用减法聚集方法产生 FIS 矩阵
	subclust	利用减法聚集方法估计簇中心

参考文献

1 楼顺天，于卫，闫华梁编著．MATLAB 程序设计语言．西安：西安电子科技大学出版社，1997

2 MATLAB user's Guide. The Mathworks, Inc. 1995

3 MATLAB Reference Guide. The Mathworks. Inc, 1995

4 SIMULINK User's Guide. The Mathworks, Inc, 1995

5 Signal Processing Toolbox User's Guide. The Mathworks, Inc, 1995

6 Ingle V K, Proakis J G. Digital Signal Processing Using MATLAB V. 4. PWA Publishing company, 1997

7 Burrus C S, etc. COmputer－Based Exercises for Signal Processing Using MATLAB. Prentice - Hall, Inc. 1994

欢迎选购西安电子科技大学出版社各类图书

书名	定价
nternet 中文网站地址簿——精彩中文网址终极推荐	25.00
中文版 Internet Explorer 4.0 套件使用大全	33.00
Intranet Ware 中文版精解	20.50
Intranet 技术及其应用	19.00
MODEM 通信编程技术	20.00
计算机网络技术	36.00
实用网络编程技术	20.50
Windows NT4.0 环境下 Intranet 组建技术	22.00
轻松使用 Microsoft FrontPage 98	9.00
互联网 Internet 和用户软件 Netscape	16.50
NetWare 3.X~4.X 实用培训教程	23.50
电子邮件、Internet 及 WWW 实用技巧	17.60
Internet 操作导航 123 问	13.00
Internet 集成浏览工具	21.00
跟我进入 Internet	22.80
Internet 资源与使用	15.80
Novell 实用网络工程方法	17.50
计算机网络	14.00
中文版 Windows 98 使用指南	26.50
Windows NT Server 4.0(中文版)组网技术	30.00
UCDOS 6.0/7.0 实用操作教程	23.00
DOS 开发环境及其高级技术	45.00
从 DOS 到 Windows	17.50
四通利方 Rich Win 4.2 操作与使用技巧	20.00
中文之星 2.0/2.5 for Windows 95 操作与使用技巧	25.00
中英文 Windows 95 快速通	24.80
中文版 Windows 95 使用大全	36.50
中文 Windows 95 环境与操作	26.00
Windows 95 使用教程	19.80
Windows 95 使用指南	22.00
Windows 3.2 快速入门	9.80
中文 Windows 3.2 实用教程	32.00
Windows 3.2/3.1 简明使用指南	18.00
中文 Windows 3.1、 3.2 配置与热点	18.50
UNIX 系统初级教程	23.00
操作系统教程	16.00
Norton 8.0 中文版操作应用技巧	31.80
微型计算机实用反病毒技术指南	16.50
DOS 工具软件例解大全	25.00
Windows 95 多媒体应用程序设计技术	27.00
微机多媒体技术及应用	18.80
多媒体电脑原理、 使用与维护	22.50
多媒体电脑安装与测试实用技术	22.80
多媒体技术览要	12.00
全国计算机等级考试(一级)试题分析与应试指南	22.50
全国计算机等级考试(二级)试题分析与应试指南 FoxBASE 语言程序设计	27.00
全国计算机等级考试(二级)试题分析与应试指南 基础部分和 C 语言程序设计	25.00
全国计算机等级考试(三级)模拟试题与解答	19.50
全国计算机等级考试(一级)模拟试题与解答	17.00
计算机等级考试培训教程(一级)	14.80
计算机应用基础教程	21.00
计算机应用办公技能培训教程	12.80
会计电算化实用教程(初级)	28.00
看图学用金蝶财务软件 for Windows	17.50
新编青少年电脑轻松起步教程	23.00
Windows 入门及其文字处理	12.80
计算机原理、 操作与文字处理(第三版)	12.80
微机操作与文字处理(修订版)	18.80
WPS 实用指南(2.0~NT1.2)	12.00
家用电脑的选购、 使用与维护	14.50
精通电脑 150 问	21.00
电脑使用 300 个怎么办?	28.60
计算机键盘练习与汉字录入技术	5.50
五笔字型速查小字典	4.50
微机常用软件英文提示信息速查手册	16.50
Office 97 中文版快学通	39.00
WPS 97 实用操作教程	20.00
中英文 Word 97 速成教程	20.50
中文 Word 8.0 快学通	20.60
中文 Excel 8.0 快学通	19.80
Photo shop 图像处理软件实用技术	16.00
3D Studio 从入门到精通	24.50
中文版 Microsoft Excel 7.0 实用教程	30.00
汉字 Lotus 1－2－3 R5 for Windows 实用教程	30.00
中文 Power Point 95 快学通	31.50
中文 Excel 95 快学通	31.50
中文 Word 95 快学通	19.50
中文 Word 6.0、 7.0 高级技巧	28.00
中文 Word 7.0 使用与提高	29.00

书名	定价
最新中文版 Word 7.0 实用教程	26.00
最新中文 Word 6.0 使用指南	21.00
高级文字处理软件 Word 6.0 应用与开发	21.50
Power Builder 4.0 使用精解	32.00
智能卡技术及应用	9.50
计算机通信技术及其程序设计	22.00
PCI 局部总线开发者指南	8.50
计算机系统安全技术与方法	31.00
自动测试技术与计算器、 仪器系统设计	19.50
实用电脑装配与硬件测试技术	21.00
常用电脑传真软硬件的安装与使用	12.80
MD110 程控数字交换机——操作维护教程	20.60
打印机疑难故障诊断与排除	17.00
打印机使用技巧与故障维修 400 例	18.00
计算机通信网原理	16.50
Visual FoxPro 5.0 中文版实用指南	22.00
图解 Visual Foxpro 5.0	23.00
关系数据库 Sybase SQL Server 应用指南	29.00
ORACLE 7 关系数据库实用技术——编程与应用	27.00
Microsoft Visual FoxPro 3.0 使用指南	45.00
Visual FoxPro 3.0b 中文版快学通	31.50
FoxPro 2.6 快速入门	16.00
FoxPro 2.6 实用教程	22.60
FoxPro 2.5、 2.6 及其程序设计	28.00
最新 FoxPro 2.6 for Windows 使用详解	20.00
FoxPro 2.5 实用程序设计与技巧	22.00
汉字 FOXBASE+及其程序设计	14.80
汉字 FOXBASE+及其程序设计——习题解答与上机指导	13.40
汉字 FOXBASE+高级程序设计技术——方法、技巧与实例	15.00
JAVA 类库及其实例大全	43.80
Visual C++ 5.0 使用指南	19.50
Borland C++ Builder 使用指南	22.50
Borland C++ Builder 编程技巧与实例	22.00
Power Builder 5.6/6.0 从入门到精通	30.00
跟我学 Quick BASIC	10.50
Visual Basic 5.0 中文版从入门到精通	24.00
Visual Basic 4.0 应用速成	24.00
Visual BASIC 3.0 for Windows 程序设计指南	21.00
Delphi 多媒体程序设计	25.00
Delphi(1.0/2.0)实用编程技术	22.50
Visual Basic 5.0 中文版实用指南	17.00
MATLAB 程序设计语言	16.80
Java 语言及其程序设计	41.00
Borlound C++ 5.0 OWL 5.0 编程技术与实例	35.50
利用 Visual C++ 2.0/4.0 编制 Windows 95 应用程序	29.50
图形用户界面设计与技术——以 Borland C++为工具(含盘)	35.00
Visual C++ for Windows 面向对象程序设计	27.50
微机科学可视化系统设计(含盘)	29.80
C 语言实用软件界面技术	15.00
C 语言实用程序荟萃	18.00
C 程序设计实用教程	14.50
C 语言实践	13.00
TURBO PASCAL 6.0 精讲、题解及应用	20.00
PASCAL 程序设计及其应用	18.00
FORTRAN 语言程序设计(第二版)	16.50
数据结构	10.00
计算方法	8.80
微型计算机原理及应用(本科)	22.00
微型机系统故障分析与实用维修	24.50
微型计算机原理与应用——以 IBM PC 系列机为例	22.00
微型计算机原理与应用(大专)(16 位)	19.00
Motorola 单片机原理及应用技术	20.50
VHDL 硬件描述语言与数字逻辑电路设计	22.00
8098 单片微型计算机应用实例	12.00
IBM PC 微机应用系统设计	14.50
IBM PC 汇编语言程序设计和接口技术	10.00
十六位微型计算机原理及接口技术	15.00
多微处理器系统设计及其实例	12.00
可编程序控制器原理及应用	23.50
计算机控制原理及其应用	23.50
PROTEL 3.31 实用精解	32.50
Auto CAD 高效机械绘图技术	29.00
AutoCAD 12.0 绘图软件包的使用与二次开发技术	27.00
Auto CAD 学与练	16.50
电子 CAD 技术基础	9.50
电子电路 CAD 技术	17.80
电子系统及专用集成电路 CAD 技术	21.50
机械 CAD/CAM 技术概论	11.50
机械 CAD 技术基础	13.20
机械 CAD 应用与开发技术	20.50
孤立子理论及其应用——光孤子理论及光孤子通信	32.80
数据融合理论与应用	20.00

书名	定价
面向对象技术	17.00
神经网络计算	27.00
神经网络应用与实现	29.00
神经网络系统理论	18.50
视觉神经系统与分布式推理理论	12.00
系统核与核度理论及其应用	6.50
实用小波分析	12.00
小波分析及其应用	7.00
UNIX 直通车	16.50
中文 Excel 95 直通车	9.00
C++ 语言直通车	11.50
数码相机的使用	9.80
摄录像技术及多媒体光盘 ----原理、使用与维修	26.50
CD · VCD · DVD----原理、选购、与维修	20.00
VCD · DVD 家庭影院	7.50
视听音响设备----原理 · 使用 · 搭配	25.00
家庭影院——组建，使用，维护	18.20
现代家庭视听指南	35.00
现代家用电器——选购、使用与维修大全	26.00
家用微波炉实用技巧	6.00
万用表检修电视机实践指南	
实用电视机维修技巧与方法	21.00
彩电摇控系统、 画中画、 有线电视加装与维修指南	22.50
电冰箱和冷柜的原理、选用与维修	14.00
空调器及其微电脑控制器的原理与维修	24.80
数字光纤通信设备(上)	19.00
数字光纤通信设备(下)	23.50
电力传动自动控制系统	30.50
集成电路速查大全	24.00
常用办公通信设备的原理、使用与维护——电话机、传真机、传印机、BB 机、大哥大	17.50
ATM 理论及应用	18.50
A/D、D/A 转换器接口技术实用线路	23.50
开关稳压电源——原理、设计与实用电路	25.00
自动控制原理(大专)	10.50
电子线路基础	17.50
录音录像技术	14.80
通讯电子线路	14.00
通信系统原理	22.00
通信基础电源	15.50
纠错码——原理与方法	28.00
电子测量技术基础	17.30
电器原理与技术(电视、 录像及家用制冷)	20.00
模拟电子技术	13.40
数字电子技术	14.90
电路分析基础(第二版)	18.00
《电路分析基础》实验与题解	12.00
网络、 信号与系统	14.50
电路基础	21.00
电路、信号与系统实验(修订版)	10.50
高频电路原理与分析(第二版)	17.80
电视原理与接收技术	12.80
英汉电脑软件词汇手册	16.80
英汉计算机操作/阅读翻译/应试词汇手册	12.50
电脑英语五周通教程	24.50
高等教育学历文凭基础英语考试指南	15.00
全国职称英语等级考试 ——模拟测试与阅读辅导(理工类)	20.00
全国职称英语等级考试 ——模拟测试与阅读辅导(综合类)	21.00
专业英语 (第一分册) 中专	16.50
专业英语 (第二分册) 中专	16.50
最新考研英语复习指导	17.80
玫瑰花与苹果树(当代英语阅读进阶 Book Ⅰ)	10.20
车轮上的学校 (当代英语阅读进阶 Book Ⅱ)	10.00
初雪(当代英语阅读进阶 Book Ⅲ)	9.00
TOEFL 语法满分技巧	26.70
大学英语常见同义词辨析	8.60
大学英语四级结构要点总汇	4.80
大学英语四级阅读综合训练	7.80
大学英语四、六级写作指导	5.00
大学英语六级模拟试题新题型精编	11.00
新编大学英语四级考试模拟题集	10.00
大学英语五、 六级研究生英语词汇手册	13.50
大学英语四级考试词汇手册	7.50
新编大学英语四级考试固定词组手册	4.00
科技英语阅读教程	17.00
科技英语语法高级教程	33.50
《线性代数》学习指导与例题分析	11.00
随机过程	13.00
最新考研数学复习指导	20.50
概率论与数理统计	8.00
高等数学(上册)	13.90
高等数学(下册)	13.50
科技英语(电子类)(大专)	14.60
高级操作系统	13.80
计算机操作系统(第二版)	19.60
计算机操作系统(第三版)	27.00

书名	定价
操作系统教程——UNIX 实例分析(第二版)	18.50
UNIX 操作系统教程	16.20
操作系统(修订版)(大专)	14.80
操作系统(中专)(第二版)	10.80
汇编语言程序设计(修订版)	21.50
《汇编语言程序设计》习题解答及实验指导	9.20
单片微机原理与应用(8098)	10.80
单片机原理及应用(51 系列)(中专)	15.50
《单片机原理及应用》学习指导	8.00
微型计算机原理(中专)	18.50
微型计算机原理(16 位机)	24.50
《微型计算机原理》学习指导书	6.50
微型计算机系统设备与维修(中专)	22.75
计算机系统结构(第二版)	19.00
计算机引论(大专)	11.20
软件系统开发技术(第二版)	12.30
数据库原理与应用	12.00
数据库原理及应用(中专)(修订版)	16.80
《数据库原理及应用》实习与实验指导	11.80
PASCAL 程序设计(大专)	12.90
PASCAL 程序设计(大专)	12.80
PASCAL 程序设计(中专)	11.00
计算机绘图	24.50
计算机通信网(修订版)	13.50
编译方法(大专)(修订版)	17.00
计算方法(大专)(修订版)	8.30
离散数学(修订版)	17.80
离散数学(大专)	12.00
管理信息系统概论(大专)	5.60
管理信息系统分析与设计	10.80
音响技术	11.80
控制电机(第二版)	15.00
自动控制基础(修订版)	14.50
工业自动化设备概论	14.00
办公自动化技术与设备	12.00
微机工业控制	12.00
短波通信	14.80
卫星通信(新版)	12.00
图像通信	12.00
扩频通信	9.80
移动通信(新版)	15.00
锁相技术(新版)	11.80
雷达原理	16.80
高频电子线路(中专)(第三版)	17.5
高频电子线路(大专)	13.00
电路分析(大专)	16.50
电工基础(中专)	15.50
数字信号处理	15.00
电视原理与现代电视系统	16.80
电视机原理与技术	15.80
电视接收技术(大专)	11.80
电视原理与接收机(中专)	15.00
天线与电波	13.80
线天线的宽频带技术	9.70
微波技术基础	15.00
电磁场有限元方法	26.80
电磁场理论基础	17.00
电磁场微波技术与天线	14.80
几何绕射理论(新版)	9.65
机构精确度	7.90
电子精密机械导论	10.90
电子机械计算机辅助设计	9.50
电子机械制造工艺学(中专)	10.90
电子工程制图(含习题集)(中专)	24.00
工程制图(含习题集)	33.20
机械基础(中专)	8.90
机械制造——实习教材(中专)	14.50
机械原理与机械零件习题册(中专)	14.75
金属切削机床(中专)	10.50
塑料模设计(中专)	16.00
模具设计与制造	16.00
工模具制造工艺学(中专)	14.70
工业企业经济活动分析(中专)	12.40
工业企业管理(中专)(修订版)	9.50
管理数学(中专)(修订版)	14.80
冲压塑压设备概论(中专)	8.00
管理心理学(中专)	7.90
公共关系学(中专)	8.20

欢迎来函索取本社最新书目和教材介绍，欢迎投稿!

从邮局或银行汇款邮购者， 款到后五天内我社将挂号发书， 加收 15%的包装邮寄费。

通信地址： 西安市太白南路 2 号　西安电子科技大学出版社发行部　邮　编： 710071

电　话：(029)8227828、8202945　传　真：(029)8213675